计算机程序设计基础（1）
02 数据类型（上）

清华大学电子工程系

杨昉

E-mail: fangyang@tsinghua.edu.cn

上节内容

- 课程基本信息介绍
- 计算机基础知识
- 编程语言的发展历史：**低级语言->高级语言**
- 程序设计的步骤，**算法的各种表示**
- C语言**编程环境的配置**
- 简单C语言程序的**编写、调试**

本节目录

2.1 计算机数据表示

- 计算机计数制
- 计算机中数的表示

2.3 基本数据类型常量

- 整型常量
- 实型（浮点型）常量
- 字符型常量

2.2 常量与变量介绍

- 常量介绍
- 变量介绍

2.4 工具推荐

- 善用搜索
- 网站
- 优秀代码风格

2.1 计算机数据表示

- 计算机计数制
  - ✓ 数制概念
  - ✓ 二进制、十进制、十六进制
  - ✓ 进制的转换
- 计算机中数的表示
  - ✓ 正负数
  - ✓ 定点数
  - ✓ 原码、反码、补码、偏移码
  - ✓ 浮点数

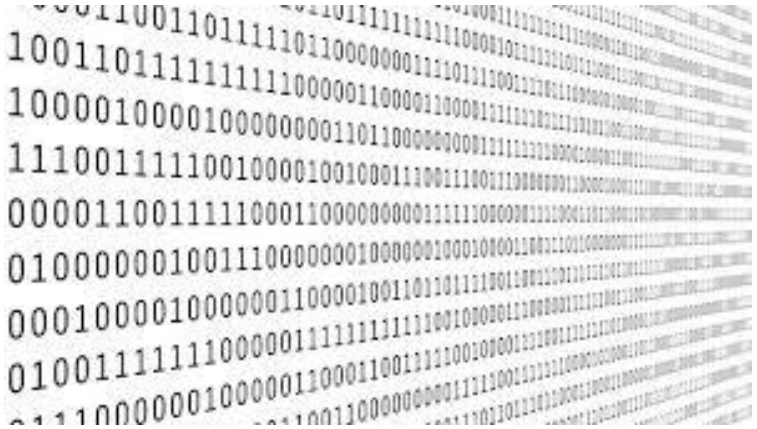
计算机数据表示：计算机计数制

- 引言: 在日常生活中，人们习惯于用十进制计数
- 一个十进制数可以用**位权**表示，通常称**某个固定位置上的计数单位为位权**
- 一个十进制数 234.13 展开成位权形式如下:

$$(234.13)_{10}$$
$$= 2 \times 10^2 + 3 \times 10^1 + 4 \times 10^0 + 1 \times 10^{-1} + 3 \times 10^{-2}$$

计算机数据表示：计算机计数制

- 在日常生活中常采用**十进制**计数，但其他进制计数也很常见
- 时间计数：采用**六十进制**，1小时60分，1分60秒
- 计算机计数：考虑到经济、可靠、容易实现、运算简便等因素，在计算机中的数都用**二进制**表示而不用十进制表示



“世界上只有10种人，一种懂二进制，一种不懂二进制。”

计算机数据表示：二进制

- 二进制数中只有两个数字符号0与1，其特点是”**逢二进一**”
- 在二进制数中，每一个数字符号(0或1)在不同的位置上具有不同的值，各位上的权值是**基数2的若干次幂**
- 优点：二进制容易表示两个状态
  - **电平：高、低**
  - **二极管：通、断**
  - **电灯：开、管**
  - **坑：有、无(VCD、DVD光盘)**
  - **逻辑：真、假**

计算机数据表示：二进制

- **除2取余法**：将十进制整数转化为二进制整数
- 例2-1：将十进制整数97转换成二进制整数
  - 将十进制数除以2，得到一个商数和一个余数
  - 再将商数除以2，又得到一个商数和一个余数
  - 继续这个过程，直到商数等于零为止
  - 每次得到的余数就是对应二进制数的各位数字
  - 注意：第一次得到的余数为二进制数的最低位
  - 最后一次得到的余数为二进制数的最高位
  - 结果为 $(97)_{10}=(1100001)_2$

2	97	
2	48	余数为1
2	24	余数为0
2	12	余数为0
2	6	余数为0
2	3	余数为0
2	1	余数为1
	0	余数为1，**商为0**

计算机数据表示：二进制

- 相反地，用**位权求和**方法可以由二进制整数得到十进制整数
- 例2-2：将二进制数$(1100001)_2$转换成十进制数

$$(97)_{10}= 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^0$$

将二进制整数的第n位乘以位权2^n，将每一位的值求和

计算机数据表示：二进制

- **乘2取整法**：将十进制小数转化为二进制小数
- 例2-3：将0.6875转换成二进制小数
  - 用2乘十进制小数，得到一个整数部分和一个小数部分；继续这个过程，直到余下的小数部分为0或满足精度要求为止
  - 最后将每次得到的整数部分从左到右排列即得到所对应的二进制小数
  - 结果为 $(0.6875)_{10} = (0.1011)_2$

	0.6875	
×	2	整数为1
	1.3750	余下小数
	0.3750	
×	2	
	0.7500	整数为0
	0.7500	余下小数
×	2	
	1.5000	整数为1
	0.5000	余下小数
×	2	
	1.0000	整数为1
	0.0000	没有余数

计算机数据表示：二进制

- 相反地，用**位权求和**方法可以由二进制小数得到十进制**小数**
- 例2-4：将二进制数$(0.1011)_2$转换成十进制数

$$(0.6875)_{10}= 1 \times 2^{-1} + 1 \times 2^{-3} + 1 \times 2^{-4}$$

将二进制小数的第n位乘以位权2^n，将每一位的值求和

计算机数据表示：二进制

- 十进制数转换二进制数的**一般方法**：为了将一个既有整数部分又有小数部分的十进制数转换成二进制数，可以将其整数部分和小数部分**分别转换**，然后再**组合起来**
- 例2-5：

整数部分：$(97)_{10}=(1100001)_2$

小数部分：$(0.6875)_{10}=(0.1011)_2$

因此：$(97.6875)_{10}=(1100001.1011)_2$

将十进制数分成整数和小数两部分，分而治之

计算机数据表示：十六进制


- 十六进制在数学中是一种“**逢16进1**”的进位制。一般用数字0到9和字母A到F表示，其中A~F表示10~15，这些称作十六进制数字
- 在十六进制数中，每一个符号在不同的位置上具有不同的值，各位上的权值是基数16的**若干次幂**
- 不同系统、编程语言对于十六进制数值有不同的表示方式：
  - C语言、C++、Shell、Python、Java语言及其他相近的语言使用字首“0x”，例如“0x5A3”
  - Intel的汇编语言中用字尾“h”来标识16进位的数
  - 其他汇编器（AT&T、Motorola）使用字首“$”，如“$5A3”

计算机数据表示：十六进制

- **除16取余法**：将十进制整数转化为十六进制整数
- 例2-6：将十进制整数986转换成十六进制整数
  - 将十进制数除以16，得到一个商数和一个余数
  - 继续这个过程，直到商数等于零为止
  - 每次得到的余数就是十六进制数的各位数字
  - 第一次得到的余数为十六进制数的最低位
  - 最后一次得到的余数为十六进制数的最高位
  - 结果为

$$(986)_{10} = (3DA)_{16}$$

16	986	余数为10（A）
16	61	余数为13（D）
16	3	余数为 3
	0	商为0，结束

计算机数据表示：十六进制

- 相反地，用**位权求和**方法可以由十六进制整数得到十进制整数
- 例2-7：将十六进制数$(3\mathrm{DA})_{16}$转换成十进制数

$$(986)_{10}= 3 \times 16^2 + 13 \times 16^1 + 10 \times 16^0$$

将十六进制整数的第n位乘以位权16^n，将每一位的值求和

计算机数据表示：十六进制

- **乘16取整法**：将十进制小数转化为十六进制小数
- 例2-8：将0.84375转换成十六进制小数
  - 用16乘十进制小数，得到一个整数部分和一个小数部分
  - 再用16乘小数部分，又得到一个整数部分和一个小数部分
  - 继续这个过程，直到余下的小数部分为0或满足精度要求为止
  - 最后将每次得到的整数部分从左到右排列即得到所对应的十六进制小数
  - 结果为 $(0.84375)_{10}=(0.D8)_{16}$

```
   0.84375
×       16
-----------
   506250
+  84375
-----------
  13.50000   整数为D
   0.50000   余下小数
×       16
-----------
   300000
+  50000
-----------
   8.00000   整数为8
   0.00000   没有余数
```

计算机数据表示：精度问题

- **精度问题**：一个十进制小数并不总是能转换为十六进制小数，根据**精度要求**只转换到小数点后某一位即可
- 例2-9：十进制小数0.32不能准确转换为十六进制小数，其小数部分为5、1、5、1...无限循环下去。如果要求取到十六进制小数点后第1位，则可以得到

$$(0.32)_{10} \approx (0.5)_{16}$$

十进制小数0.32不能准确地转换为十六进制小数。在这种情况下，可以根据精度要求取到十六进制小数点后的某一位为止，最后得到的只是近似的十六进制小数

计算机数据表示：十六进制

- 相反地，用**位权求和**方法可以由十六进制小数得到十进制**小数**
- 例2-10：将八进制数$(0.D8)_{16}$转换成十进制数

$$(0.84375)_{10}= 13 \times 16^{-1} + 8 \times 16^{-2}$$

将十六进制小数的第n位乘以位权16^n，将每一位的值求和

计算机数据表示：十六进制

- 十进制数转换十六进制数的**一般方法**：为了将一个既有整数部分又有小数部分的十进制数转换成十六进制数，可以将其整数部分和小数部分**分别转换**，然后再**组合起来**

- 例2-11

整数部分：$(986)_{10}=(3\mathrm{DA})_{16}$

小数部分：$(0.84375)_{10}=(0.\mathrm{D8})_{16}$

因此：$(986.84375)_{10}=(3\mathrm{DA}.\mathrm{D8})_{16}$

将十进制数分成整数和小数两部分，分而治之

计算机数据表示：计数制转换

- 不同计数制的关系

	二进制	八进制	十进制	十六进制
基数	2	8	10	16
位权	2^K	8^K	10^K	16^K
数字符号	0~1	0~7	0~9	0~9与A~F

计算机数据表示：计数制转换

十进制	二进制	八进制	十六进制
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

计算机数据表示：符号位的表示

- 在计算机中，一个数的正、负号是用一个二进制位来表示。一般将整个二进制数的**最高位**定为二进制数的**符号位**
- 符号位为“0” 时表示**正数**，符号位为“1”时表示**负数**
- 以8位二进制数为例
  - 如果用8个二进制位表示一个**无符号的整数**，由于不考虑数的符号问题，该8位都可以用来表示数值，因此，8个二进制位可以表示的**最大无符号数**为255(即8位全是“1”)
  - 如果用8个二进制位表示一个**有符号的整数**，由于最高位为符号位，具体表示数值的只有7位。在这种情况下，所能表示的数值**范围为-127～127**

计算机数据表示：符号位的表示

- 以16位二进制数为例
  - 如果用16个二进制位表示一个**无符号的整数**，由于不考虑数的符号问题，该16位都可以用来表示数值，因此，16个二进制位可以表示的最大无符号数为**65535**(即16位全是1)
  - 如果用16个二进制位表示一个**有符号的整数**，由于最高位为符号位，具体表示数值的只有15位。在这种情况下，所能表示的数值范围为**-32767～32767**
- 例2-12：十进制数+513和-513用16位二进制数表示分别为

$$(+513)_{10} = (0000001000000001)_2 = (0201)_{16}$$
$$(-513)_{10} = (1000001000000001)_2 = (8201)_{16}$$

计算机数据表示：定点整数的表示

- 所谓定点数是指**小数点位置固定的数**
- 在计算机中，通常用定点数来表示**整数**与纯小数，分别称为**定点整数**与**定点小数**
- **定点整数**
  - 在定点整数中，一个数的最高二进制位是**符号位**，用以表示数的符号
  - 小数点的位置默认为在最低(即最右边)的二进制位的后面，但小数点**不单独占**一个二进制位
  - 因此，在一个定点整数中，符号位右边的所有二进制位数表示的是一个**整数值**

计算机数据表示：定点小数的表示

- 所谓定点数是指**小数点位置固定的数**
- 在计算机中，通常用定点数来表示整数与**纯小数**，分别称为**定点整数**与**定点小数**
- **定点小数**
  - 在定点小数中，一个数的最高二进制位是**符号位**，用以表示数的符号
  - 小数点的位置默认为在符号位的后面，它也**不单独占**一个二进制位
  - 因此，在一个定点小数中，符号位右边的所有二进制位数表示的是一个**纯小数**

计算机数据表示：原码

- 原码
  - 所谓原码就是前面所介绍的**二进制定点数表示**
  - 即原码的**符号位在最高位**，0表示正，1表示负，数值部分按一般的二进制形式表示
- 例2-13：

$(+50)_{10}$的8位二进制原码为$(00110010)_{原}$

$(-50)_{10}$的8位二进制原码为$(10110010)_{原}$

$(+33)_{10}$的8位二进制原码为$(00100001)_{原}$

计算机数据表示：原码

- 原码
  - 在二进制原码中，使用的二进制**位数越多**，所能表示的数的**范围就越大**
  - 一般来说，如果用n位二进制来存放一个定点整数的原码，能表示的整数值范围为：

$$-2^{n-1}+1 \sim 2^{n-1}-1$$

  - 问题：重复表示了-0和+0

计算机数据表示：反码

- 反码
  - **正数**的反码和原码**相同**
  - **负数**的反码是对该数的**原码除符号位外各位取反**(即将0变为1，1变为0)
- 例2-14: $(+50)_{10}$的8位二进制反码为$(00110010)_{反}$

  $(-50)_{10}$的8位二进制反码为$(11001101)_{反}$

  $(-33)_{10}$的8位二进制反码为$(11011110)_{反}$

  则有 $(00000000)_{反}$所表示的十进制数为0

  $(11111111)_{反}$所表示的十进制数为-0

计算机数据表示：补码

- 补码
  - 正数的补码和原码相同
  - 负数的补码是在该数的反码的最后(即最右边)一位上加1
  - 解决了-0的表示问题，$(10000000)_{补}$所表示的十进制数为-128
  - 一般来说，如果用n位二进制来存放一个定点整数的补码，能表示的整数值范围为:

$$-2^{n-1} \sim 2^{n-1}-1$$

  - 一个数的补码的补码就是这个数的原码本身

计算机数据表示：偏移码

- 偏移码
  - 不管是正数还是负数，其**补码的符号位取反**即是**偏移码**
  - 由此可知，定点数用偏移码表示后，其最高位也为符号位
  - 但符号位的取值刚好和原码与补码相反，1表示正，0表示负；而其**数值部分与相应的补码相同**
  - 一般来说，如果用n位二进制来存放一个定点整数的偏移码，能表示的整数值范围为：

$$-2^{n-1} \sim 2^{n-1}-1$$

计算机数据表示：偏移码

- 偏移码
  - $(+33)_{10}$的8位二进制补码为$(00100001)_{补}=(10100001)_{偏移码}$
  - $(-50)_{10}$的8位二进制补码为$(11001110)_{补}=(01001110)_{偏移码}$
  - $(+0)_{10}$的8位二进制补码为$(00000000)_{补}=(10000000)_{偏移码}$
  - $(-128)_{10}$的8位二进制补码为$(10000000)_{补}=(00000000)_{偏移码}$
  - $(+127)_{10}$的8位二进制补码为$(01111111)_{补}=(11111111)_{偏移码}$

计算机数据表示：偏移码

- 偏移码
  - 定点数用偏移码表示后，也可以执行加减运算，必须将结果的**符号位取反后**才是偏移码形式的结果
- 例2-15：$(-50)_{10}$的8位二进制偏移码为$(01001110)_{偏移码}$

  $(+33)_{10}$的8位二进制偏移码为$(10100001)_{偏移码}$
  - 其结果$(11101111)_{偏移码}$所表示的十进制数**并不是-17**
  - -17的偏移码为$(01101111)_{偏移码}$

$$\begin{array}{rl} & 01001110 \\ + & 10100001 \\ \hline & 11101111 \end{array}$$

$(-50)_{10}$的二进制偏移码

$(+33)_{10}$的二进制偏移码

计算机数据表示：偏移码

- 偏移码
  - $(-50)_{10}+(+33)_{10} = (01001110)_{偏移码}+(10100001)_{偏移码}$

    $= (01101111)_{偏移码}$ **符号位取反**

    $= (10010001)_{原}$

    $= (-17)_{10}$

符号位为0的偏移码转原码：**各位求反（包括符号位），末位加1**

不管是正数还是负数，其**补码的符号位取反**即是**偏移码**

计算机数据表示：计算机中数的表示

- 小结：原/反/补/偏移码
  - 在二进制定点数的四种表示中，**原码**比较直观，但**不能用于具体运算**
  - **补码与偏移码**可用于具体**运算**
  - **反码**只起到由原码转换为补码或偏移码的**中介作用**

课堂练习1

- 为什么**补码的补码就是原码**?
  - 若$(x)_{原}=(y)_{补}$，$(y)_{原}=(z)_{补}$，则 x=z;
  - 正数：补码和原码相同
  - 负数：原码求补码“**取反加一**”，补码求原码“**减一取反**”
  - 负数不考虑符号：反码等于 $2^{N-1}-1$ 减去原码（N为总位数）
  - 负数不考虑符号：补码等于2^{N-1}减去原码（N为总位数）

$$x+y=2^{N-1},\ y+z=2^{N-1}$$

  - 得x=z，对于二进制“**取反加一**”**等价于**“**减一取反**”

课堂练习2

- **偏移码**是对谁的偏移?
  - 不管是正数还是负数，其**补码的符号位取反**即是**偏移码**
  - $(+0)_{10}$的8位二进制补码为$(00000000)_{补}=(10000000)_{偏移码}$
  - $(-128)_{10}$的8位二进制补码为$(10000000)_{补}=(00000000)_{偏移码}$
  - $(+127)_{10}$的8位二进制补码为$(01111111)_{补}=(11111111)_{偏移码}$
  - 以无符号数表示**相对于-2^{N-1}的偏移**，令x是原码**数值部分**
  - 正数：符号位取反为1，数值部分与相应补码相同；**偏移码**$x+2^{N-1}$
  - 负数：符号位取反为0，数值部分$2^{N-1}-x$；**偏移码**$-x-(-2^{N-1})$

课堂练习3

- 采用**二进制补码**计算$(+33)_{10}-(+50)_{10}$和$(+33)_{10}+(+50)_{10}$
  - 首先对于第一个式子，其等价于

$$(+33)_{10}-(+50)_{10}=(-50)_{10}+(+33)_{10}$$

  - 而$(-50)_{10}$的二进制补码可以表示为$(\mathbf{11001110})_2$
  - $(+33)_{10}$的二进制补码可以表示为$(\mathbf{00100001})_2$
  - $(11101111)_2$为补码，原码是$(10010001)_2$，对应十进制数为-17

$$\begin{array}{rll} & 11001110 & (-50)_{10}\text{的二进制补码} \\ + & 00100001 & (+33)_{10}\text{的二进制补码} \\ \hline & 11101111 & \end{array}$$

课堂练习3

- 对于第二个式子，其等价于

$$(+33)_{10}+(+50)_{10}=(00100001)_{补}+(00110010)_{补}$$
$$=(01010011)_{补}$$
$$=(01010011)_{原}$$
$$=(+83)_{10}$$

正数的补码与原码相同

- 补码使得正负数的加减法和普通数一样而不必考虑符号位,因此计算机中**使用补码存整数**，以便于计算

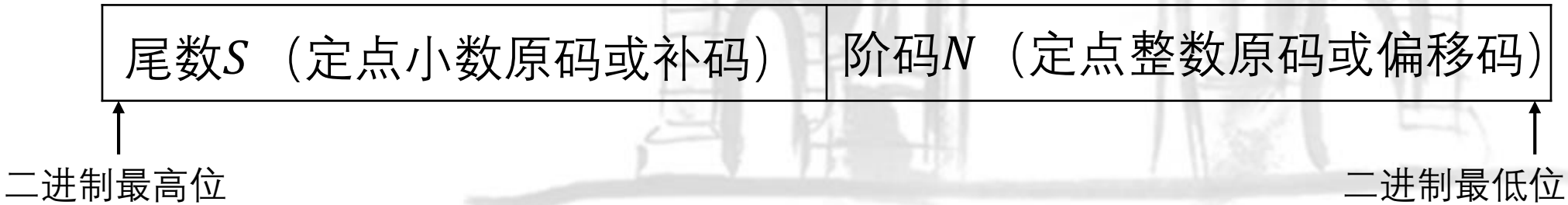
$$\begin{array}{rl} & 00100001 \\ + & 00110010 \\ \hline & 01010011 \end{array}$$

$(+33)_{10}$的二进制补码

$(+50)_{10}$的二进制补码

计算机数据表示：尾数和阶码

- 一个**既有整数部分又有小数部分**的十进制数R可以表示为$R = Q \times 10^n$，其中Q为一个小数，n为一个整数
- 对于既有整数部分又有小数部分的二进制数P也可以表示为$P = S \times 2^N$，其中S为一个**二进制定点小数**，被称为P的**尾数**；N为一个二进制定点整数，称为P的**阶码**（反映了二进制数P的小数点的实际位置）
- 在计算机中，通常用**一串连续的二进制位**来存放二进制浮点数



计算机数据表示：尾数和阶码

- 例2-16：用16位二进制定点小数补码以及8位二进制定点整数补码表示十进制数-254.75
  - 首先将$(-254.75)_{10}$转化为二进制数，即

    $$(-254.75)_{10} = (-11111110.11)_2 = 2^8 \times (-0.111111011)_2$$

  - 然后将**尾数**化成16位二进制定点小数，即

    $$S = (-0.111111011)_2 = (1111111101100000)_2$$

    小数点位置

  - 其反码为 $S = (1000000010011111)_{反}$
  - 其补码为 $S = (1000000010100000)_{补} = (80A0)_{16}$

计算机数据表示：尾数和阶码

- 例2-16：用16位二进制定点小数补码以及8位二进制定点整数补码表示十进制数-254.75
  - 将**阶码**8转换为二进制数，即 $(+8)_{10}=(+1000)_2$
  - 化成8位二进制定点整数为，即 $N=(+1000)_2=(00001000)_2$

小数点位置

  - 其补码为（**正数的补码是原码本身**）

$$N=(00001000)_{补}=(08)_{16}$$

2.2 常量与变量

- 常量与变量介绍
- 变量定义
  - ✓ 整型数
  - ✓ 实型数
  - ✓ 字符型数
  - ✓ 空类型及其他类型
- 基本数据类型介绍

常量与变量：常量与变量介绍

- **常量**
  - 在程序执行过程中其值不变的量（不能改变的量）
- **变量**
  - 在程序执行过程中其中的值可以改变的量

*“一个程序的运行过程，实际上是在处理各种各样的**数据**。一般来说，在程序中数据是以变量或者常量的形式表示”*

常量与变量：基本数据类型介绍

- 整型
  - 有符号基本整型(int或signed int)
  - 无符号基本整型(unsigned int)
  - 有符号长整型(long或signed long)
  - 无符号长整型(unsigned long)
  - 有符号短整型(short或signed short)
  - 无符号短整型(unsigned short)
  - 有符号超长整型(long long或_int64)
- 实型
  - 分为单精度实型(float)与双精度实型(double)

常量与变量：基本数据类型介绍

- 字符型
  - 有符号字符型（char或signed char）
  - 无符号字符型（unsigned char）
- 空类型
  - void类型
- 除了上述四种基本数据类型外，C语言还有
  - 枚举类型
  - 构造类型
  - 指针类型等复合数据类型

常量与变量：变量定义

- C语言规定：
  - 程序中的每一个变量都有一个**唯一的名字**，称为**变量名**
  - 变量在使用前必须**首先定义**
  - 所谓定义一个变量，就是系统根据变量的数据类型为该变量**分配存储空间**
  - **变量名即代表其存储空间**，以便在程序执行过程中在这个存储空间中存取数据

常量与变量：变量命名

- 变量名的命名规则：
  - 变量名必须以字母或下划线**开头**，后面可以跟**若干个**字母、数字或下划线
  - 变量名区分大小写字母。例如：_a，_A，a123，A123__
  - 不同的编译系统对变量名中的字符总个数有不同的规定
  - 常用的变量命名法有**匈牙利命名法**、**驼峰命名法（小）**、**下划线命名法**、**帕斯卡命名法（大驼峰式）**等，可以参考教程 https://zhuanlan.zhihu.com/p/89909623

常量与变量：变量命名

- 变量命名举例：
  - **小驼峰式**命名法：要求第一个单词首字母小写，后面其他单词首字母大写
  - 例如定义一个变量来记录我的年龄，用小驼峰方式定义为

    int myAge
  - **大驼峰式**命名法：要求每个单词的第一个字母都要大写，用大驼峰式命名法则为

    int MyAge
  - **下划线**命名： int my_age

2.3 基本数据类型常量

- 整型常量
  - ✓ 整型分类
  - ✓ 整型表示
- 实型常量
  - ✓ 十进制表示
  - ✓ 指数表示（科学计数法）
- 字符型常量
  - ✓ 转义字符

基本数据类型常量：整型常量

- 整型常量的分类
  - 有符号与无符号**基本整型常量**
  - 有符号与无符号**长整型常量**
  - 有符号与无符号**短整型常量**

常量类型		存储空间	数值范围
有符号	短整型常量	2B	$-32768 \sim 32767(-2^{15} \sim 2^{15}-1)$
	基本整型常量	4B	$-2147483648 \sim 2147483647(-2^{31} \sim 2^{31}-1)$
无符号	短整型常量	2B	$0 \sim 65535(2^{16}-1)$
	基本整型常量	4B	$0 \sim 4294967295(2^{32}-1)$

注： B代表字节(Byte)

基本数据类型常量：整型常量

- 整型常量的表示
  - 十进制表示：可以使用的符号有十个数字符号0～9以及+与-
    - 对于正整数，前面的“+”号可以省略
    - 对于长整型常量，一般要在其后加一个英文字母L或l
    - 对于无符号整型常量，一般要在其后加一个英文字母U或u
    - 对于64位的超长整型，一般要在其后加i64
  - 十六进制表示：整型常量以0x或0X开头，符号有0～9与A～F(或a～f)
  - 八进制表示：整型常量以0(零)开头，可以使用的符号有0～7

基本数据类型常量：整型常量

- 例2-17
  - 123u、123L、123i64数值是否相同？有什么区别？

✓数值相同

✓123u是**无符号基本整型常量**，在计算机中用4个字节存放

✓123L是**长整型常量**，在计算机中也要用4个字节存放

✓123i64是**64位超长整型常量**，在计算机中要用8个字节存放

基本数据类型常量：实型常量

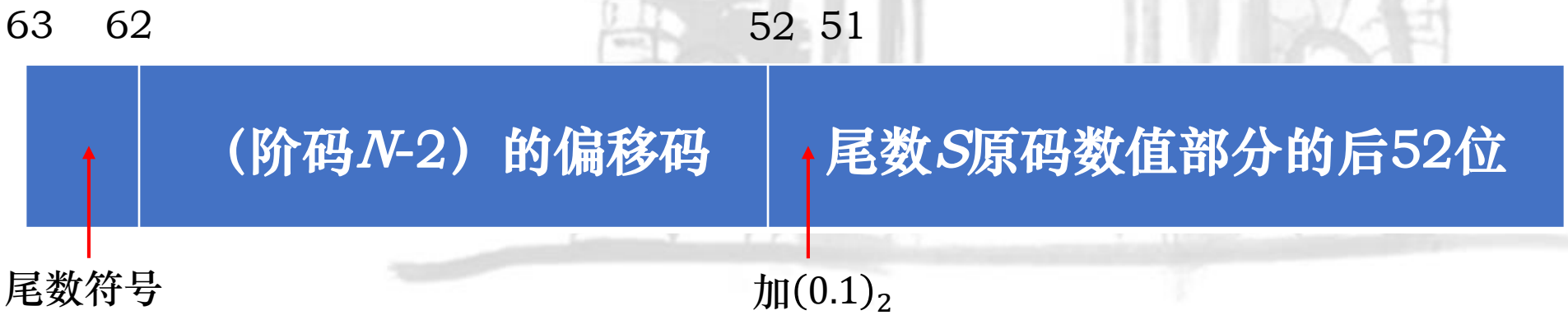
- 十进制表示
  - 包括符号“+”与“-”，0～9十个数字以及小数点“.”
- 指数形式(科学记数法)
  - 包括符号“+”与“-”，0～9十个数字，小数点“.”以及e(或E)
  - 其中e（或E）前面必须要有数字，后面必须为整数

基本数据类型常量：实型常量

- 计算机中的表示形式

$$P = S \times 2^N$$

  - 尾数S：二进制定点小数
  - 阶码N：二进制定点整数，反映了二进制数P的小数点的实际位置
  - 存储形式（IEEE 754 标准格式）



基本数据类型常量：实型常量

- 例2-18：十进制实型数97.6875在计算机中表示形式

$$(97.6875)_{10} = (1100001.1011)_2$$

$$= (0.11000011011)_2 \times 2^7$$

则尾数S = $(01100001101100000000000000000000$
$00000000000000000000000000)_{\text{原码}}$

- **(阶码$N-2$) = (10000000101)$_{\text{偏移码}}$**
- 用十六进制表示为40 58 6C 00 00 00 00 00，共占8个字节（64位）

基本数据类型常量：实型常量

- 计算机系统分配给一个数据的存储空间是有限的
- 一般来说，一个实型常量**无法转换**成与之**等值**的有限位的二进制数据
- 其有限位以后的数字将被舍去，由此就会产生**舍入误差**
  - 舍入误差是指运算得到的近似值和精确值之间的差异
  - 舍入误差是**量化误差**的一种形式
  - 在某些情况下，误差会随着运算次数增加而**积累得很大**，最终得出没有意义的运算结果

基本数据类型常量：舍入误差

- 舍入误差是运算得到的**近似值和精确值之间的差异**
- 例2-19：下列C程序的功能是将10个实型数0.1进行累加，然后将累加结果输出

```
1  #include <stdio.h>
2  void main(){
3      int k;            /*定义整型变量k*/
4      double x, z;      /*定义双精度是型变量x与z*/
5      z = 1.0;          /*实数1.0赋给变量z*/
6      x = 0.0;
7      for (k = 0; k < 10; k++)
8          x = x + 0.1;      /* 10个0.1累加到变量x中*/
9      printf("z= %20.17f\n", z);   /*输出变量z的值*/
10     printf("x= %20.17f\n", x);   /*输出变量x的值*/
11 }
```

精确值

```
z = 1.00000000000000000
x = 0.99999999999999999
请按任意键继续……
```

近似值

基本数据类型常量：字符型常量

转义字符	含义
'\n'	换行
'\r'	回车（不换行）
'\b'	退格
'\t'	制表(横向跳格)
'\''	单引号(单撇号)
'\"'	双引号(双撇号)
'\ddd'	1～3位八进制数所代表的ASCII码字符，最大'\377'
'\xhh'	1～2位十六进制数所代表的ASCII码字符，最大'\xFF'
'\f'	走纸换页
'\\'	反斜杠字符
'\a'	响铃一次（等价于'\007'）

基本数据类型常量：字符型常量

- 说明
  - 字符常量： 'A' ，'\n', '0', '\0'，占**一个字节**。单个字符用**单引号**引起来
  - 其中\为**转义符**，使字符表示别的意思
    - '\n'不再表示字符'n'，而是表示回车换行
    - '\t'不再表示字符't'，而是表示制表符Tab
    - '\0'不再表示字符'0'，而是表示字符串结束符，其ASCII内码为0
    - '\''表示单引号字符
    - '\ddd'为八进制的ASCII字符。最大'\377'
    - '\xdd'为十六进制的ASCII字符，最大'\xFF'
    - '\r'回车不换行，'\a'响铃一次（等价于'\007'）

基本数据类型常量：字符型常量

- 注意
  - 'ab' 是错误的，单引号中只能是单个字符
  - 但'\xab'正确，表示十六进制数
  - '377'是错误的，但'\377'是正确的
    - 这里377为八进制数，转换为十进制表示255
    - 因此'\377'表示ASCII字符表的最后一位，是一个保留字符

课堂练习4

- 用C语言编写上节课件中例1-1：计算并输出z=y/x
  - 第一步：输入x与y
  - 第二步：**判断x是否为0**
    - 若x=0，则打印错误信息
    - 否则计算y/x->z并输出z
- 第一步：**引入头文件（第8讲）**
  - 引入了你想直接调用的系统函数，如打印函数printf

如何编写一个C程序?

第3讲

课堂练习4

- 第二步：**写函数**
  - 入口是你的main函数，代码写在main里面即可
  - 未来还将学习定义新的函数，并在main里面调用（第7、8讲）
- 第三步：**定义变量（本讲）**
  - C语言一定要遵循先定义后使用原则，如整型为int
- 第四步：**常见语法（后续）** 第5讲 第6讲
  - 后续将学习C语言常用语法，如if、for、while等

如何编写一个C程序?

课堂练习4

●C语言编写上节课件例1-1

```
1  #include <stdio.h>                  //包含头文件stdio.h
2  void main(){                        //定义main函数，这是程序的主体
3    float x,y,z;                      //定义float型变量x,y,z
4    printf("input x,y:");             //屏幕上打印信息
5    scanf("%f %f",&x,&y);             //输入x与y的值，注意其中细节
6    if(x==0)                          //若x=0，则打印错误信息
7      printf("error!x=0\n");          //"\n"表示换行
8    else{                             //否则计算z，一个模块里多个语句则用{}括起来
9      z = y/x;                        //
10     printf("z = %f\n", z);          //打印z的值
11   }
12 }
```

（头文件，第8讲）

（函数，第7讲）

（数据类型，本讲）

（输入输出，第3讲）

（if顺序结构，第5讲）

（分支结构命令，第5讲）

（输出，第3讲）

实际演示：解决warning C4996

- 当在高版本VS中使用scanf函数时，会出现以下警告
  - VS已经不支持**strcpy**函数，认为是不安全的，可能会产生诸如**内存泄露**、**缓冲区溢出**、**非法访问**等安全问题

```
warning C4996: 'scanf': This function or variable may be unsafe. Consider using scanf_s instead.
```

- 解决这一警告的方法是在程序开头加上如下说明:

```
#pragma warning(disable : 4996)
```

2.4 工具推荐

- 善用搜索
- 网站
- 优秀代码风格

学习方法——借鉴和学习优秀资料

- 编程中遇到难题，学会主动寻找“老师”——**搜索引擎**
- 在论坛上和全世界优秀的代码工程师交流
- 一些成熟的辅助编程工具，可以帮助你更快、更专业地掌握编程技巧

- 习近平：今天，我们要铸就中华文化新辉煌，就要以更加博大的胸怀，更加广泛地开展同各国的文化交流，**更加积极主动地学习借鉴世界一切优秀文明成果**

工具推荐：善用搜索

- 编程遇到问题可以在**百度、Google、Bing等网站**进行搜索。例如不明白数组的赋值操作，可以进行如下搜索

- 对于中文搜索不出来的答案，尝试使用**英文搜索**，很多时候答案更加全面和正确

工具推荐：编程网站

- 代码问题解答
  - 中文网站比较权威的有

    CDSN https://www.csdn.net/

    编程中国https://www.bccn.net/

    博客园https://www.cnblogs.com/

    脚本之家https://www.jb51.net/等
  - 英文网站比较权威的有
    - Stack Overflow https://stackoverflow.com/
    - CodeProject https://www.codeproject.com/
    - StackExchange https://stackexchange.com/等

工具推荐：开源网站

- 代码下载和保存
  - 对于想要阅读他人优秀代码，或者为自己代码建库的同学，可以使用一些托管网站，下面是一些开源的网站

    Github https://github.com/

    GitLab https://gitlab.com/

    Gitolite https://gitolite.com/gitolite/index.html

    Gitea https://gitea.io/en-us/
  - 在本地管理自己的代码版本（当代码量上去后版本管理很重要），可以使用Git： https://git-scm.com/

工具推荐：编程习惯

- 编程练习
  - 一些良好的代码题库

    Leetcode https://leetcode-cn.com/

    洛谷 https://www.luogu.com.cn/

    HackerEarth https://www.hackerearth.com/ 等

对于各种凌乱的电脑问题，手机问题，其他行业的人，以为程序员们什么都会；程序员中，一般程序员以为技术好的程序员什么都会；技术好的程序员，都在网上苦苦找答案 ……

（最好老师：搜索引擎）

工具推荐：代码风格

- 代码风格错误示范

```
                                                  extern int
                                                      errno
                                                        ;char
                                                           grrr
                                ;main(                      r,
  argv, argc )                int    argc                    ,
   r       ;                 char *argv[];{int                 P( );
#define x  int i,            j,cc[4];printf("      choo choo\n"     ) ;
x  ;if    (P(  !             i               )       |  cc[  !      j ]
&  P(j    )>2  ?             j               :       i  ){*  argv[i++ +!-i]
;              for    (i=                  0;;    i++                    );
_exit(argv[argc- 2    / cc[1*argc]|-1<<4 ]    ) ;printf("%d",P(""));}}
  P  (    a  )   char a   ;  {    a  ;   while(    a  >       "  B   "
  /* -    by E            ricM    arsh             all-      */);    }
```

```
                                        #define r(R) R"()"
                                /*[*/#include  /**/<stdio.h>
                              #include<math.h>/*!![crc=0f527cd2]*/
                            float I,bu,k,i,F,u,U,K,O;char o[5200];int
                          #define R(U) (sizeof('U')==1||sizeof(U"1"[0])==1)
                         h=0,t=-1,m=80,n=26,d,g,p=0,q=0,v=0,y=112,x=40;  float
                        N(float/*x*/_){g=1<<30;d=-~d*1103515245&--g;return  d*_
                       /g;}void/**/w(int/**/_){if(t<0){for(g=0;g<5200;o[g++   ]=
                       0);for(;g;o[g+79]=10)g-=80;for(t=37;g<62;o[80+g++]=32)   ;
                      }if(m&&o[h*80+m-1]==10){for(g=0;g<79;o[t*80+g++]=0){}o[t
                      ++*80+g]=10;t%=64;n+=2;I=N(70)+5;if(n>30&&(I-x)*(I-x)+n*
                     n>1600&&R()){O=0;F=(x=0x1!=sizeof(' '))?k=1+N(2),i=12-k+N(
                     8),N(4):(k=17+N(5),i=0,r()[0]?O=.1:  0);for(u=U=-.05;u<32;
                     U=k+i+i*.5*sin((u+=.05)+F))for( K=0   ;K< U;K+=.1)if((bu=K*
                    sin(u/5),g=I+cos( u/5) *K)>=0&&g  <     79  )o[g+(int)(t+44+
                    bu*(.5-(bu>0?3*O:  O)   ) )%64*  80      ]  =32;x*=02// */2
                   -1;n=O+x?n=I+(x?0   :N     (k)-   k          /2),g=(t+42  )%
                   64,m=-~g%64,x?g=m          =-~       m%64:0  ,n>5?o[g*80   +
                  n-3]=o[m*80+n-3]=       0:   0              ,n <75?o[g*80+n
                  +2]=o[m*80+n+2]=0   :0:0;                     x=I;}h=-~h%64
                 ;m=0;}putchar((g=o [h*                         80+m++])?g:_);
                if(g){w(_);}}void W                              (const char*_
               ){for(;*_;w(*_++));}                              int main(int a
               ,char**_){while(a--)d              +=_[a          ]-(char*)0;W( \
              "#include<stdio.h>typed"             "e"        "f\40int\400;v"
              "oid o(0 _){putchar(_);}0"                   "\40main(){0"  ""
             "*_[512],**p=_,**d,b,q;for(b=0;b"       "++<512;p=_+q)_[q"    \
             "=(p-_+1)*9%512]=(0*)p;") ;      for(;(g= getchar())-EOF;p=
             q){q=p;for(v=512;p-q-g&&q-p-              g;  v--)q=-~q*9%512
             ;W("o(");if(p>q)w(y),w(45);w(                    40);w(y^=20
             );w(075);for(a=0;a<v;a++)w(42);                   for(W("(0**"
              );a--;w(42)){}w(41);w(y^024);w(                   41);if(p<=q)w(
               45),w(y^20);W(");");}for(a=7;a-6                   ;W(a<6?"{;}":""
                  ))for(a  =0;a  <6 &&   !o[h*80+m                  +a];a++){}W("r"
                    "etu"  /*J   */      "rn+0;}\n"                        );return
                       /*                  "#*/0                              ;}
```

工具推荐：优秀代码风格

- C语言代码风格
  - Google的风格
    https://google.github.io/styleguide/cppguide.html
    https://zh-google-styleguide.readthedocs.io/en/latest/google-cpp-styleguide/
  - Linux内核风格
    https://www.kernel.org/doc/Documentation/process/coding-style.rst
  - 一份完善的总结资料
    https://www.zhihu.com/column/c_1336772315556458496

工具推荐：优秀代码风格

- 以**代码注释**为例，参考Google给出的风格指南
  - 可以使用//或者/**/进行注释（//常见），最好全文统一
  - 文件注释：在编写的每一个文件开头加入版权公告，例如由xxx于xxxx年xx月xx日完成的；并且要注明该文件的内容，例如是测试了一个类还是编写了一个函数
  - 函数注释：在函数声明处**注释函数的功能**，并且使用叙述式（“Opens the file”）而非指令式（“Open the file”），包含的内容有函数的输入输出、函数是否分配了必须由调用者释放的空间、参数是否可以为空指针等

工具推荐：优秀代码风格

- 以**代码注释**为例，参考Google给出的风格指南
  - 变量注释：通常**变量名**本身足以很好说明**变量用途**. 某些情况下，也需要额外的注释说明；对于全局变量，所有**全局变量**也要注释说明含义及用途，以及作为全局变量的原因
  - 实现注释：对于代码中巧妙的，晦涩的，有趣的，重要的地方加以注释。其中巧妙或复杂的代码**段前要加注释**；比较隐晦的地方要**在行尾加入注释**（在行尾**空两格**进行注释）；如果函数参数的**意义不明显**，也需要加入注释

工具推荐：优秀代码风格

- 以**代码注释**为例，参考Google给出的风格指南
  - 在代码编写中，不要描述**显而易见的现象**，要假设读代码的人 C++ 水平比你高，即便他/她可能不知道你的用意
  - 注意**标点**、**拼写**和**语法**；写的好的注释比差的要易读的多
  - 注释要**言简意赅**，不要拖沓冗余

```
DoSomething();                  // Comment here so the comments line up.
DoSomethingElseThatIsLonger();  // Two spaces between the code and the comment.
{ // One space before comment when opening a new scope is allowed,
  // thus the comment lines up with the following comments and code.
  DoSomethingElse();  // Two spaces before line comments normally.
}
std::vector<string> list{
                    // Comments in braced lists describe the next element...
                    "First item",
                    // .. and should be aligned appropriately.
"Second item"};
DoSomething(); /* For trailing block comments, one space is fine. */
```

本课总结

- **计算机数据表示**
  - 计算机计数制：数制概念介绍；2/8/10/16进制；进制转换
  - 计算机中数的表示：正负数；定点数；原/反/补/偏移码；浮点数
- **常量与变量介绍**
  - 常量与变量介绍
  - 基本数据类型介绍：整型、实型、字符型、空类型，以及枚举、指针等
  - 变量定义：使用前先定义；命名规则；匈牙利命名法
- **基本数据类型常量**
  - 整型常量：整型常量的分类与表示
  - 实型常量：实型常量的十进制与指数型表示
  - 字符型常量：转义字符
- **工具推荐**

本课作业

- 第二讲作业
  - 教材第二章习题1，2，4
  - 完成后将**word文档或拍照**提交到网络学堂

- 附加题：

  有能力的同学可以尝试测试如下样例
  - 给定有符号短整型数a=32767，b=-32768;输出a+1和b-1的值，观察输出的结果并分析原因


THANKS