第1篇:C/C++ 内存中的数据表示

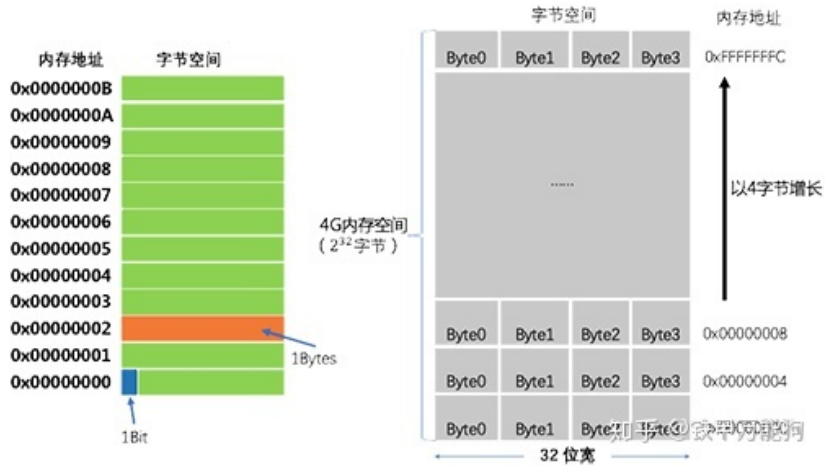
铁甲万能狗
自由开发者，专攻C++/Python后端开发(简书平台同号)

创作声明：内容包含虚构创作

这篇文章假设你已经基本了以下的基础

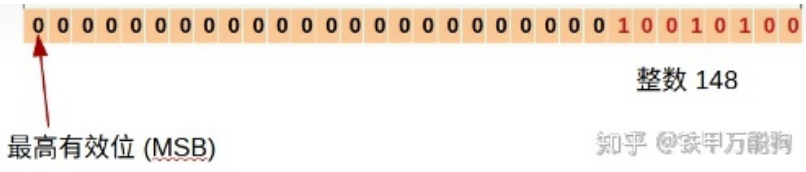
- C的基本数据类型有了一定的了解
- 了解基本的进制转换算法

这是C/C++内存管理系列文章的第一篇,主要讲述C中基本的数据类型如何在内存中表示.
RAM是计算机的主要或主要内存。 它是执行程序时存储文本，数据，指令和中间结果的地方。 总存储器被组织成字节数，每个字节再次被分成8位。 位是存储器中的最小单元，这些位是将数据存储为1和0的位置，称为二进制数据。 在存储器中，每个字节用一个称为地址的数字标识。 它总是一个正数。



计算机内存寻址模型

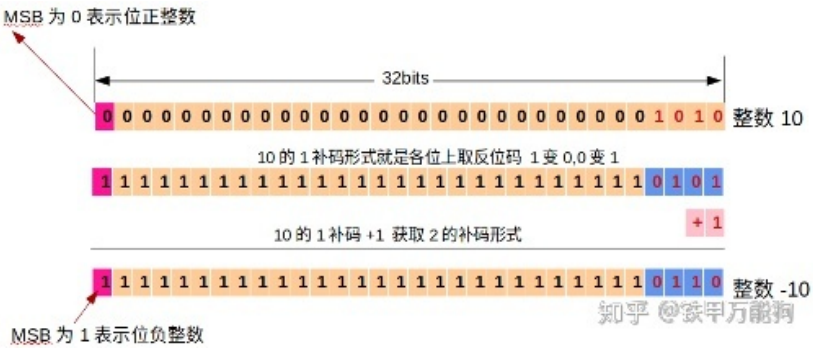
在执行程序时，总内存被分为多个段，称为文本，未初始化的全局，初始化的全局，堆栈和堆段。 将整个程序加载到文本段中，并在堆栈存储器中选择存储器到变量.
备注:关于堆栈的内存话题,这个会另行写文，这里就不展开了
内存中的整数表示：
例如，当我们将数字148分配给变量x时，数字148首先转换为其相等的二进制10010100，然后存储到存储器中。 即使使用浮点数也会发生同样的事情。
在C/C++中如果声明一个signed int类型的便来唔那么在内存中就会启用MSB(最高有效位)位用于指示该数字是正数还是负数。
对于正数，MSB将为0
对于负数，MSB将为1
在我们的例子中，148为正，所以MSB为0。
这个等效的148的二进制文件将存储在32位内存中，如下所示，



负整数如何存储在内存中?
计算机使用特殊机制来存储负数，这是2的补码格式。
在开始之前，让我们应该要知道1的数字补码

- **1的数字补码**：1的数字补码只是反转实际数字的二进制位。
  那么以整数10为例,它的二进制是1010,

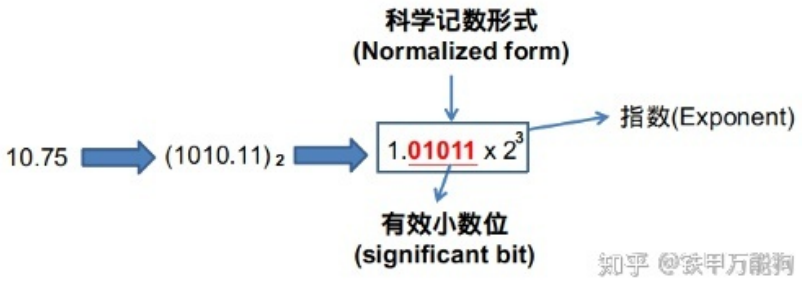
1. **10的1补码形式"0101"(仅是各位上0切换到1,1切换到0)**
2. **获取2补码形式只需加1到1之后得到实数补码即可**



以上该2的补码形式就表示负数-10,存储在指定的内存空间中

内存中的浮点数表示
要存储浮点数，将在计算机中分配4字节（32位）内存。
1位用于MSB标记正/负
指数部分为8位
有效小数位是23位
十进制的浮点数转换二进制形式,这里用一个简单的示例step by step讲解

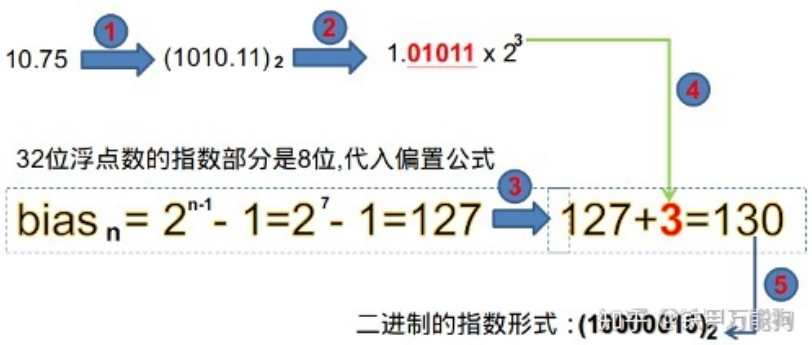
1. **浮动数字将转换为二进制数字,例如我们将10.75转换为二进制形式1010.11**
2. **将转换后的二进制数转换成科学记数法的形式**,具体如下步骤,小数点左移后小数点左边保留1位二进制数,小数点右边的是**有效小数位(significant bit)**



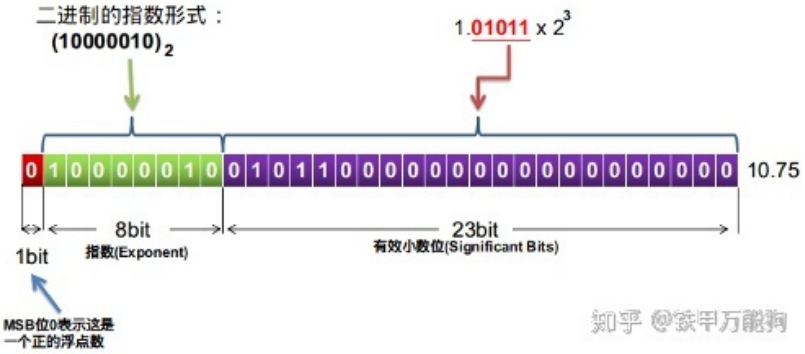
1. **向指数添加偏差**
   在浮点数中，并不像有符号整数那样有在**2的补码形式**来存储负数。为了表明这一点，设计计算机的

$$bias_n = 2^{n-1} - 1$$

这里，由于32位的浮点数为指数分配了8位, 所以n将是8,那么套公式,2的7次幂减1就是127.
因此,科学记数形式中的指数,就会是**当前指数**+**偏置值**,即3+127=130,然后将130转化为二进制形式就是**10000010**



将标记0置号，因为10.75是正数
指数值为130，即（10000010）2
有效小数位是1.01011，这里我们可以将小数点之前消除1，因为无论我们总是二进制科学记数化形式为1.???*2^n这种形式.所以,不需要存储1.只需存储小数点后面 **有效小数位**,这个例子就是01011



内存中的double类型的数据表示
要存储double类型的数据需要分配8字节(64位)内存空间。

- 1位用于MSB
- 11位为**指数**
- 52位表示**有效小数位**

double和float表示之间的唯一区别是**偏置值**。这里我们使用11位表示指数,十进制转换二进制的算法都是一样,没必要麻烦

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

数据来源于网络
与数字数据一样，即使字符不能按原样存储，因为计算机只知道二进制数系统,它也会首先转换为相等的二进制数，然后存储到存储器的位中。 键盘上的每个字符都具有相等的二进制值。 等于该二进制值的十进制数称为ASCII（美国信息交换标准代码）值。 比如说字符'A'的二进制值是01000001，十进制等于65.所以'A'的ASCII值是65.为了存储字符值，计算机将分配1字节(8位)内存.

字符字面量在内存中的表示

备注:这里我并不会详细谈论更复杂的字符集

发布于 2020-08-14 21:01

内存（RAM） 数据类型 C / C++

文章被以下专栏收录

C/C++内存管理
侧重C/C++内存模型，反编译分析