Представление чисел в компьютере

<table>
<tr><th colspan="6">Числа</th></tr>
<tr><th colspan="5">Целые</th><th>Вещественные</th></tr>
<tr><th>Без знака</th><th colspan="4">Со знаком</th><td rowspan="3"></td></tr>
<tr><td rowspan="2">Прямой код</td><th>Положительные</th><th colspan="3">Отрицательные</th></tr>
<tr><td>Прямой код = Дополнительный код = Обратный код</td><td>Прямой код</td><td>Дополнительный код</td><td>Обратный код</td></tr>
</table>

Целые числа без знака
(используется прямой код)

Обычно занимают в памяти компьютера один или два байта.

1 байт

3

0	0	0	0	0	0	1	1

2 байта

3

0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Целые числа со знаком
(используется дополнительный код)

В К-разрядной целочисленной компьютерной арифметике $2^К \equiv 0$.

К-разрядный дополнительный код отрицательного числа X – это запись в К разрядах положительного числа $2^К – |X|$, где $|X|$ – модуль числа X, $|X| \leq 2^{К–1}$

1 байт

3

0	0	0	0	0	0	1	1

знак

Число

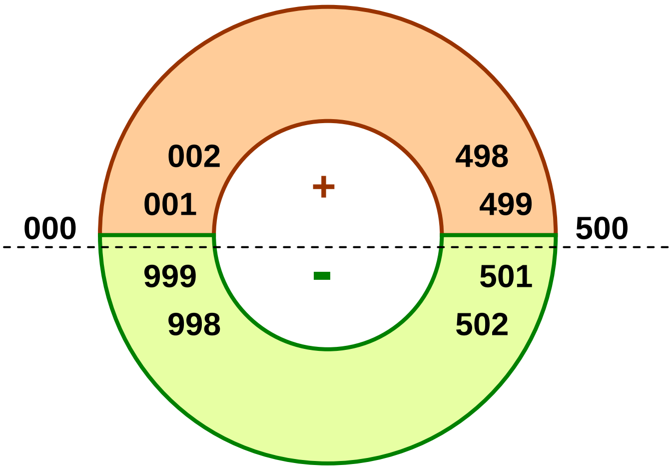
Число

2 байта

-3

1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1

Дополнительный код



– 2 дополнительный код 2 + 998 = 1000

Разрядная сетка

	8	7	6	5	4	3	2	1	
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	1	+1
	0	0	0	0	0	0	1	0	+1
							…		
	0	1	1	1	1	1	1	1	+1
	1	0	0	0	0	0	0	0	+1
	1	0	0	0	0	0	0	1	+1
							…		
	1	1	1	1	1	1	1	1	+1
1	0	0	0	0	0	0	0	0	

Правила представления отрицательных чисел в памяти компьютера

1	Определить длину (в битах), которую занимает число в памяти компьютера	Пусть длина числа – 8 бит.								
2	Записать число без знака	$-3 \rightarrow 3$								
3	Перевести в двоичную систему счисления	**прямой код**	0	0	0	0	0	0	1	1
4	Заменяются все 1 на 0, а 0 на 1	**обратный код**	1	1	1	1	1	1	0	0
5	Прибавляется 1 к последнему разряду	обратный код	1	1	1	1	1	1	0	0
										1
		дополнительный код	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**

Как компьютер выполняет арифметические операции над целыми числами

Арифметические операции		
	Сложение	
	Вычитание	Сложение обратных или дополнительных кодов уменьшаемого и вычитаемого
	Умножение	Последовательность сложений и сдвигов
	Деление	Многократное прибавление к делимому дополнительного кода делителя

Операция вычитания

	Человек	Компьютер
5	0 0 0 0 0 1 0 1	0 0 0 0 0 1 0 1
–	–	+
3	0 0 0 0 0 0 1 1	1 1 1 1 1 1 0 1
2	0 0 0 0 0 0 1 0	1 0 0 0 0 0 0 1 0

	Компьютер
3	0 0 0 0 0 0 1 1
–	+
5	1 1 1 1 1 0 1 1
– 2	1 1 1 1 1 1 1 0

Проверка

Дополнительный код	1	1	1	1	1	1	1	0
								1
Обратный код	1	1	1	1	1	1	0	1
Прямой код	0	0	0	0	0	0	1	0
Исходное число								– 2

Операция умножения

		Компьютер								
5	101	0	0	0	0	0	0	0	0	**Множитель**
x							1	0	1	**101101**
45	101101	0	0	0	0	0	1	0	1	Сдвиг на две позиции
225	11100001				1	0	1			**101100**
		0	0	0	1	1	0	0	1	Сдвиг на одну позицию
				1	0	1				**101000**
		0	1	0	0	0	0	0	1	Сдвиг на две позиции
		1	0	1						**100000**
		1	1	1	0	0	0	0	1	

Вещественные числа

Вещественные числа в компьютере хранятся в формате с **плавающей запятой**, который опирается на нормализованную форму записи чисел.

Нормализованная запись числа

При представлении **целых чисел** в компьютере ограничением может служить лишь величина записываемого числа (количество бит, которое отведено под число).

При записи **вещественного числа** – точность представления, т.е. количество значащих цифр, которое удается сохранить в ограниченном числе разрядов.

Пример.

Допустим, у нас есть калькулятор, в котором на экране дисплея для вывода чисел есть только 10 знакомест, включая знак числа, запятую между целой и дробной частями.

Если нам необходимо работать с числами

–6234000000	11 знакомест
–623,4	6 знакомест
–0,0000006234	13 знакомест

то на дисплее нашего калькулятора можно будет отобразить число –623,4.

Эта задача может быть решена, если представить числа:

–6.234E+09	$-6.234\cdot10^{9}$
–6.234E+02	$-6.234\cdot10^{2}$
–6.234E–07	$-6.234\cdot10^{-7}$

Такая запись отражает **экспоненциальную форму** записи чисел.

Любое число **A** в экспоненциальной форме представляется в виде

$$A = \pm\, m \times Q^{P}$$

Q – основание системы счисления,

m – мантисса числа,

P – порядок числа

Пример.

Число 123,45 в экспоненциальной форме записи можно представить:

$123.45 \cdot 10^{0}$

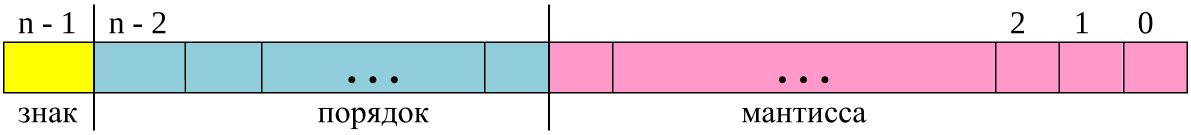
$12.345 \cdot 10^{1}$

$1.2345 \cdot 10^{2}$

$12345 \cdot 10^{-2}$

Погрешность числа

В компьютерном представлении вещественных чисел максимально допустимое количество цифр в мантиссе определяет точность, с которой может быть представлено число.



Пример.

Допустим, у нас есть калькулятор, в котором на экране дисплея для вывода чисел есть только 10 знакомест, включая знак числа, запятую между целой и дробной частями.

Исходное число	Число на экране дисплея калькулятор	После перевода с фиксированной точкой	Абсолютная погрешность	Относительная погрешность
248,53786	+2.485E+02	248,5	0,03786	
2485378600,0	+2.485E+09	2485000000	378600	
0,00024853786	+2.485E–04	0,0002485	0,00000003786	

Абсолютная погрешность представления числа X – модуль разности между значением числа и неким его представлением (компьютерным, калькуляторным):

$$\varepsilon = |X - Xa|$$

Несмотря на то, что в абсолютном исчислении погрешность может быть значительно больше 1, относительно величины самого числа ее порядок остается неизменным.

Относительная погрешность представления числа X – модуль разности между значением числа и неким его представлением (компьютерным, калькуляторным):

$$\varepsilon = |(X - Xa)/X|$$

Исходное число	Число на экране дисплея калькулятор	После перевода с фиксированной точкой	Абсолютная погрешность	Относительная погрешность
248,53786	+2.485E+02	248,5	0,03786	0,00015233
2485378600,0	+2.485E+09	2485000000	378600	0,00015233
0,00024853786	+2.485E–04	0,0002485	0,00000003786	0,00015233

Что дает знание величин абсолютной и относительной погрешностей?

Абсолютная погрешность показывает, насколько полученный результат отличается от истинного результата. При решении реальных задач знание этой величины позволяет оценить, **насколько достоверный результат был получен**.

Относительная погрешность показывает, **сколько верных старших цифр** содержит результат.
Относительная погрешность 0,00015233 означает, что мы имеем три безусловно верные значащие цифры результата.
Относительная погрешность непосредственно связана с количеством разрядов, отводимых для представления мантиссы нормализованного числа.

> В компьютерной записи вещественных чисел с плавающей запятой **количество цифр, отводимых под запись порядка**, определяет, насколько большие и насколько маленькие положительные числа могут быть представлены.

Пример.

Допустим, у нас есть калькулятор, в котором на экране дисплея для вывода чисел есть только 10 знакомест, включая знак числа, запятую между целой и дробной частями, порядок имеет две цифры.

Самое большое число: +9.999E+99

Самое маленькое число: +1.000E−99

Представление вещественных чисел в формате с плавающей запятой

Справедливы и следующие равенства:

$$25{,}324 = 2{,}5324\times10^{1} = 0{,}0025324\times10^{4} = 2532{,}4\times10^{-2} \text{ и т.п.}$$

Чтобы не было неоднозначности записи чисел в компьютере, используют нормализованное представление числа в форме с плавающей точкой.

Нормализованная запись отличного от нуля вещественного числа – это запись числа

$$A = \pm\, m \cdot Q^{P}$$

m – правильная дробь, у которой первая цифра после запятой не равна нулю, т.е. $1/Q \le m < 1$,

P – целое число (положительное, отрицательное или ноль)

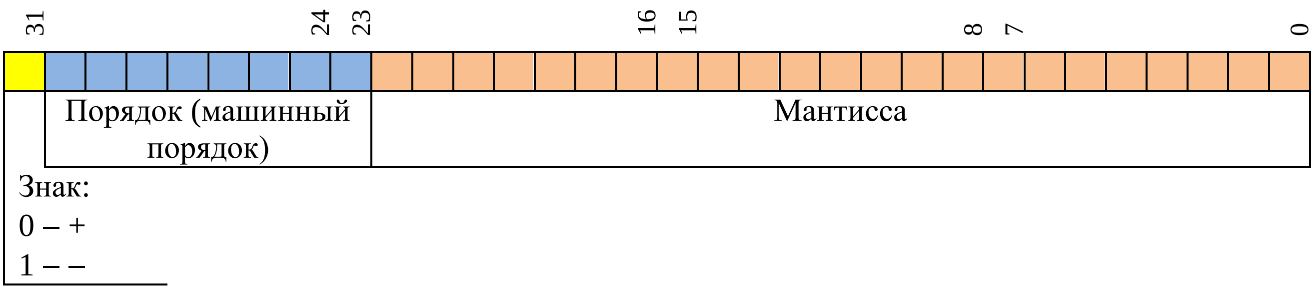
Мантисса меньше единицы и первая значащая цифра – не ноль. Для рассмотренного числа нормализованным представлением будет: 0.25324×10^{2}.

Ноль не может быть записан в нормализованной форме.

Запись нуля является нормализованной, если и мантисса, и порядок равны нулю.

0.0×10^{0}

4 байта – Single



В отведенное в памяти поле в соответствии с типом числа записываются мантисса, характеристика и знак числа.

При этом необходимо отметить следующее:

1. для чисел типа Real характеристика хранится в младшем байте памяти, для чисел других типов - в старших байтах;
2. мантисса всегда хранится в **прямом коде**;
3. знак числа находится всегда в старшем бите старшего байта;
4. целая часть мантиссы (для нормализованного числа всегда 1) для чисел типа Real, Single, Double не хранится (является скрытой). В числах типа Extended все разряды мантиссы хранятся в памяти ЭВМ.

Машинный порядок

В семи двоичных разрядах помещаются двоичные числа в диапазоне от 0000000 до 1111111. Всего 128 значений. В десятичной системе это соответствует диапазону от 0 до 127.
Знак порядка в ячейке не хранится. Но порядок, очевидно, может быть как положительным, так и отрицательным. Разумно эти 128 значений разделить поровну между положительными и отрицательными значениями порядка. В таком случае между машинным порядком и истинным (назовем его математическим) устанавливается следующее соответствие:

Математический порядок	0	1	2	3	…	126	127
Машинный порядок	–64	–63	–62	-61		62	63

Если обозначить машинный порядок МР, а математический – Р, то связь между ними выразится формулой:

$$MP = P + 64$$

Итак, машинный порядок смещен относительно математического на 64 единицы и имеет только положительные значения. При выполнении вычислений с плавающей точкой процессор это смещение учитывает.

И, наконец, в двоичной системе:

$$MP_2 = P_2 + 100\ 0000_2$$

Правила формирования машинного представления вещественного числа в памяти ЭВМ

1	Определить тип числа (Real, Single, …)	Число бит для хранения мантиссы и порядка зависит от типа вещественного числа.
2	Число представляется в двоичном коде	**$25{,}324_{10} = 11001{,}0101001011110001101_2$**
3	Двоичное число нормализуется и определяется порядок числа	Для чисел, больших единицы, плавающая точка переносится влево, определяя положительный порядок. Для чисел, меньших единицы, точка переносится вправо, определяя отрицательный порядок. **$0{,}110010101001011110001101 \times 10^{101}$** Здесь мантисса, основание системы счисления ($2_{10}=10_2$) и порядок ($5_{10}=100_2$) записаны в двоичной системе. Сдвигается число на один разряд влево так, чтобы в целой части была 1: **$1{,}10010101001011110001101 \times 10^{100}$**
4	Определяется мантисса	Отбрасывается первая (старшая) единица **1 10010101001011110001101**
5	Определяется машинный порядок	**$MP = 100_2 + 100\ 0000_2 = 100\ 0100_2$**
6	Окончательный результат	**01000100 | 10010101 | 00101111 | 00011010** Знак числа — Мантисса

Если число было бы отрицательным, например, –**$25{,}324_{10}$**, то оно выглядело бы так:

11000100	**10010101**	**00101111**	**00011010**

Знак числа