Представление чисел в ЭВМ

Все числовые данные хранятся в памяти компьютера в двоичном виде, т. е. в виде последовательностей нулей и единиц, однако формы хранения целых и вещественных чисел **различны.**

Как Вы считаете, почему это так?

Необходимость различного представления целых и вещественных чисел вызвана тем, что скорость выполнения операций над целыми числами существенно выше, чем над вещественными числами.

Текстовая, графическая, звуковая информация, количество деталей, акций, сотрудников – эти и многие другие данные выражаются **целыми числами**.

Для решения математических и физических задач, в которых невозможно обойтись только целыми числами, используются **вещественные числа**.

Границы представления целых чисел

Целые числа могут быть представлены как **беззнаковые** - только неотрицательные, и как **знаковые** – положительные и отрицательные.

В зависимости от количества разрядов ячейки памяти границы представления целых чисел будут различными.

Разрядность	**8**	**16**	**32**
Минимум (без знака)	**0**	**0**	**0**
Максимум (без знака)	**255**	**65 535**	**4 294 967 295**
Минимум (со знаком)	**- 128**	**- 32 768**	**- 2 147 483 648**
Максимум (со знаком)	**127**	**32 767**	**2 147 483 647**

Почему диапазоны представления знаковых и беззнаковых целых чисел различны?

Представление целых чисел

Целые числа, как знаковые, так и беззнаковые, хранятся в формате с **фиксированной точкой**.

При таком представлении чисел все разряды ячейки, кроме знакового, если он есть, служат для изображения разрядов числа.

Причем каждому разряду ячейки соответствует один и тот же разряд числа. Именно поэтому такое представление называется с фиксированной точкой, так как **фиксируется место десятичной точки** перед определенным разрядом.

Для целых чисел десятичная точка находится после младшего разряда, то есть **вне разрядной сетки**.

Форматы представления целых чисел

При представлении беззнаковых чисел все разряды ячейки отводятся под представление разрядов самого числа.

Минимальное 0	0	0	0	0	0	0	0	0
Максимальное 255	1	1	1	1	1	1	1	1

В случае представления знаковых целых чисел старший (левый) разряд ячейки отводится под хранение знака числа. В этот разряд заносится 0, если число положительное и 1 – если число отрицательное. Поскольку для хранения разрядов самого числа количество разрядов ячейки уменьшается на единицу, границы представления уменьшаются в два раза.

Почему минимальное знаковое число в 8-разрядной ячейке –128, а максимальное +127?

Прямой код числа

Представление в форме «знак» - «величина», когда старший разряд ячейки отводится под знак, называется **прямым кодом** двоичного числа.

Число **1001_2**	0	0	0	0	1	0	0	1
Число **-1001_2**	1	0	0	0	1	0	0	1

Положительные числа в ЭВМ всегда представляются с помощью прямого кода. Прямой код числа полностью совпадает с записью самого числа в ячейке памяти машины. Прямые коды соответствующих положительных и отрицательных чисел отличаются только значением старшего разряда ячейки.

Но отрицательные целые числа представляются в ЭВМ с помощью совсем другого кода, который называется **дополнительным кодом**.

Почему используется дополнительный код числа?

Например, запись числа 243 в одном байте будет выглядеть так:

Число **243**

1	**1**	**1**	**1**	**0**	**0**	**1**	**1**

Но если эту запись рассматривать как запись числа со знаком, значением записи будет число - 115

Число **-115**

1	**1**	**1**	**1**	**0**	**0**	**1**	**1**

Подобное обстоятельство в значительной мере осложняет алгоритмы действий с целыми числами, имеющими разные знаки.

А как Вы думаете, в чём состоит усложнение алгоритмов?

Дополнительный код числа

Дополнительный код положительного числа равен прямому коду этого числа. Например, прямой и дополнительный коды двоичного числа **1001_2** для 8-разрядной ячейки равны **00001001**.

Дополнительный код отрицательного числа **m** равен **2^k-|m|**, где **k** – количество разрядов в ячейке, а **$|m| < 2^k$.**

Другими словами, **дополнительный код** отрицательного числа – это **дополнение |m| до 2^k.**

Если **k=8, $|m|=01100101_2$**, то **дополнительный код** можно получить как разность **$100000000_2 – 01100101_2 = 00011011_2$.**

Или **$01100101_2 + 00011011_2 = 100000000_2$ (155+101=256)**

В чем состоит сложность получения дополнительного кода?

Алгоритм получения дополнительного кода отрицательного числа

Для получения дополнительного **k-разрядного** кода отрицательного числа необходимо:

1) модуль числа представить прямым кодом в **k** двоичных разрядах;
2) значения всех битов **инвертировать**: все нули заменить на единицы, а единицы – на нули (таким образом получается **k**-разрядный **обратный код** исходного числа);
3) к полученному **обратному коду**, трактуемому как k-разрядное неотрицательное двоичное число, **прибавить единицу**.

Пример 1. Получение восьмиразрядного дополнительного кода числа –52:
00110100 – число |-52|=52 в прямом коде;
11001011 – число –52 в обратном коде;
11001100 – число –52 в дополнительном коде.

В какой последовательности следует инвертировать значения битов прямого кода числа?

Задание. Получите дополнительный код числа –52 в шестнадцати разрядах.

Нормализованная запись чисел

Для представления вещественных чисел принят способ представления с **плавающей точкой**. Этот способ опирается на **нормализованную запись** действительного числа.

Определение. Нормализованной называется запись отличного от нуля действительного числа в виде $\mathbf{m \bullet P^q}$, где **q** – целое число (положительное, отрицательное или ноль), а **m** – правильная **P**-ричная дробь, у которой первая цифра после запятой не равна нулю, т. е. $\mathbf{1/P \le m < 1}$. При этом **m** называется мантиссой числа, **q** – порядком числа.

Пример 2. Примеры нормализации чисел.

1) $3.1415926=0.31415926 \bullet 10^1$

2) $1000=0.1 \bullet 10^4$

3) $-0.123456789= -0.123456789 \bullet 10^0$

4) $0.0000107_8=0.107_8 \bullet 8^{-4}$

5) $1000.0001_2=0.10000001_2 \bullet 2^4$

6) $-0.0001101_2 = -0.1101_2 \bullet 2^{-3}$

Запись нуля считается нормализованной, если и мантисса, и порядок равны нулю, т. е. $\mathbf{0 = 0.0 \bullet 10^0}$

Объясните, что и когда «плавает» в форме представления чисел с «плавающей точкой»?

Компьютерное представление вещественных чисел

Как и для целых чисел, при представлении вещественных чисел используется двоичная система счисления, поэтому предварительно число должно быть переведено в двоичную систему.

При представлении чисел с плавающей точкой в разрядах ячейки отводится место для знака числа, знака порядка, абсолютной величины порядка, абсолютной величины мантиссы.

знак числа (-)

абсолютная величина порядка (13)

1	0	00001101	101101100001011110010

знак порядка (+)

абсолютная величина мантиссы (5826486)

В ячейке записано отрицательное двоичное число –1011011000010.111100110

В десятичном представлении это будет число –5826.486

Объясните, чем определяются точность вычислений и допустимый диапазон представимых чисел?

Особенности арифметических операций над числами с плавающей точкой

Предположим для простоты, что в ячейке памяти один десятичный разряд порядка и пять десятичных разрядов мантиссы.

Сложение. Пусть необходимо найти сумму $10^{2} \bullet 0.23619 + 10^{-2} \bullet 0.71824$
Перед сложением (и вычитанием) производится выравнивание порядков.
При этом число с меньшим порядком преобразуется.

$$10^{2} \bullet 0.23619 + 10^{2} \bullet 0.0071824 = 10^{2} \bullet 0.23690824$$

Но для записи мантиссы имеются только пять ячеек, поэтому полученная восьмиразрядная сумма округляется до пяти разрядов - $10^{2} \bullet 0.23691$, при этом точность результата теряется. Вычитание производится аналогично.

Умножение. При умножении двух чисел с плавающей точкой их порядки надо просто сложить, а мантиссы – перемножить без выравнивания порядков. Результат при необходимости округляется.

Деление. При делении из порядка делимого вычитается порядок делителя, а мантисса делимого делится на мантиссу делителя. При этом может произойти и переполнение порядка, и потеря точности мантиссы частного.

Как Вы считаете, операции над числами с плавающей точкой – это операции над целыми или над вещественными числами?

Контрольные вопросы

1. Как будут представлены в 8-битном знаковом типе числа:
а) –1; б) –10; в) –120; г) –102;

2. Запишите следующие двоичные числа в прямом, обратном и дополнительном коде для 8-разрядной ячейки:
а) –1000; б) –11101; в) –1; г) –1111111;

3. Приведите к нормализованному виду числа, оставляя их в тех же системах счисления, в которых они записаны:

а) $-0.\ 000001011101_2$; б) 987654321_{10}; в) 100.01_2; г) $-0.\ 001502_8$;

4. Запишите в естественной форме с фиксированной запятой следующие нормализованные числа:

а) $0.\ 1011_2 \bullet 2^1$; б)$0.\ 1011_2 \bullet 2^{11}$; в)$0.\ 12345_{10} \bullet 10^{-3}$; г) $-0.\ 40065_8 \bullet 8^{-4}$;