Основы основ

Счет

Счет в областях, связанных с алгоритмами, программированием и информатикой, как правило, начинается с 0, а не с 1

Обычный счет	1	2	3	4	5	6	7	8	9
Программирование	0	1	2	3	4	5	6	7	8

Тогда очевидно, что последний элемент будет иметь номер N – 1, где N – количество элементов

Почему возникает такая необходимость будет рассмотрено позже

Массивы

Массивом называется набор элементов, который имеет заданный размер (количество элементов), а каждый элемент имеет свой индекс (порядковый номер).

Индекс	0	1	2	3	4	5	6	7	8
Элемент	1	2	4	8	16	32	64	128	256

Например, это массив степеней двойки (массив размера 9)

Системы счисления

Одни и те же числа могут быть представлены используя N различных знаков. Такой способ представления называется (позиционной) системой счисления по основанию N, или же N-ичной системой счисления.

- Наиболее используемые:
- Двоичная (0, 1)
- Десятичная (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
- Шестнадцатеричная (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

Bin	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111	10000	10001
Dec	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11

Стоит отметить, что нумерация в системах счисления соответствует алгоритмическому счету – с 0, а не с 1

Биты и байты

Память в электронике, как правило, состоит из устройств, способных находиться в одном из двух состояниях. Тогда говорят, что такая ячейка памяти способна хранить 1 бит информации (либо 0, либо 1). Если объединить 8 бит, то получится байт.

	Бит	Байт
-	1 бит	1 Б = 8 бит
Кило	1 Кбит = 1024 бит	1 КБ = 1024 Б
Мега	1 Мбит = 1024 Кбит	1 МБ = 1024 КБ
Гига	1 Гбит = 1024 Мбит	1 ГБ = 1024 МБ
Пета	1 Пбит = 1024 Гбит	1 ПБ = 1024 ГБ

Очевидно, что количество значений, которое может принимать ячейка в N бит, равно 2^N.

Например, байт может принимать одно из $2^8 = 256$ состояний.

Память как массив бит

В электронике, в частности, в компьютерах, память представляется как массив байт, индекс которого называют адресом или смещением (offset) в памяти. Байт принято представлять в шестнадцатеричном виде, тогда он имеет вид двух цифр в этой системе счисления (00 - FF).

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
0009B100	75	73	00	6B	69	6C	6C	73	65	72	76	65	72	00	73	63
0009B110	72	69	70	74	55	73	61	67	65	00	73	74	72	69	6E	67
0009B120	55	73	61	67	65	00	73	61	79	00	74	65	6C	6C	00	00
0009B130	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0009B140	63	68	61	6C	6C	65	6E	67	65	52	65	73	70	6F	6E	73
0009B150	65	20	25	69	20	25	69	00	63	68	61	6C	6C	65	6E	67
0009B160	65	52	65	73	70	6F	6E	73	65	20	25	69	00	63	6F	64
0009B170	75	6F	61	75	74	68	6F	72	69	7A	65	2E	61	63	74	69
0009B180	76	69	73	69	6F	6E	2E	63	6F	6D	00	52	65	73	6F	6C
0009B190	76	69	6E	67	20	25	73	0A	00	43	6F	75	6C	64	6E	27
0009B1A0	74	20	72	65	73	6F	6C	76	65	20	61	64	64	72	65	73
0009B1B0	73	0A	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0009B1C0	25	73	20	72	65	73	6F	6C	76	65	64	20	74	6F	20	25
0009B1D0	69	2E	25	69	2E	25	69	2E	25	69	3A	25	69	0A	00	61
0009B1E0	75	74	68	6F	72	69	7A	65	20	73	65	72	76	65	72	20

Побитовые операции

	Конъюнкция (И) A & B	Дизъюнкция (ИЛИ) A | B	Исключающее ИЛИ A ^ B	Отрицание ~A
Операнд A	0110 0010	0110 0010	0110 0010	0110 0010
Операнд B	1110 1000	1110 1000	1110 1000	
Результат	0110 0000	1110 1010	1000 1010	1001 1101

	Побитовый сдвиг влево A << B	Побитовый сдвиг вправо A >> B
Операнд A	0110 0010	0110 0010
Операнд B	1	1
Результат	1100 0100	0011 0001

Арифметические операции с целыми числами

	Сложение A + B	Вычитание A - B	Умножение A * B	Деление A / B
Операнд A	5	5	5	5
Операнд B	10	10	10	10
Результат	15	-5	50	0

Взятие остатка обозначается как A % B или A mod B.
При делении некоторого числа A на B, первое можно представить в виде суммы $A = B \cdot C + R$, где C - частное, а R - остаток, все числа целые.
Таким образом, A % B = A mod B = R. Приведем примеры:

4 % 2 = {4 = 2 · 2 + 0} = 0
3 % 2 = {3 = 2 · 1 + 1} = 1
5 % 10 = {5 = 10 · 0 + 5} = 5
101 % 10 = {101 = 10 · 10 + 1} = 1

Алгебра логики. Основные операции Связь с побитовыми операциями

Логика оперирует состояниями истина (1) и ложь (0). При этом, обычные числа преобразуются по логике 0 – ложь, все остальное (в т.ч. < 0) – истина.

X	Y	X & Y	X | Y	X ^ Y	!X
0	0	0	0	0	1
0	1	0	1	1	
1	0	0	1	1	0
1	1	1	1	0	

А теперь посмотрим еще раз на побитовые операции, обращая внимание на биты операндов A и B, стоящих на одних и тех же позициях:

	Конъюнкция (И) A & B	Дизъюнкция (ИЛИ) A | B	Исключающее ИЛИ A ^ B	Отрицание ~A
Операнд A	0110 0010	0110 0010	0110 0010	0110 0010
Операнд B	1110 1000	1110 1000	1110 1000	
Результат	0110 0000	1110 1010	1000 1010	1001 1101

Целые числа

- Очень сложно работать, имея на руках только 256 значений. Потому из двух вариант формируется то, что на сегодняшний день принято называть словом (WORD, $2^{16} = 65536$), из четырех – двойное слово (DWORD, $2^{32} = 4294967296$), а из 8 – четверное слово (RWORD, QWORD, 2^{64}).

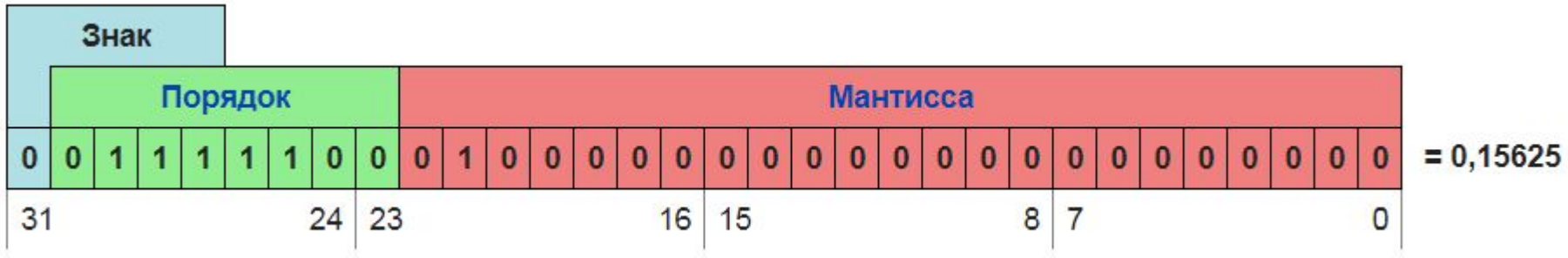
Разрядность оперируемых чисел ограничена разрядностью регистров процессора.

Отрицательные числа представляются так называемым дополнительным кодом, который получается при инвертировании битов «прямого» числа и прибавлением единицы.

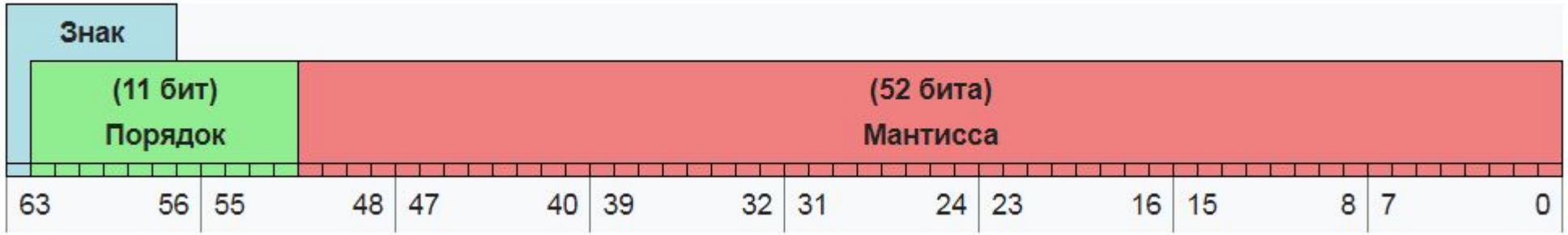
2	00000010
1	00000001
0	00000000
-1	11111111
-2	11111110

Вещественные числа

- Любое конечное вещественное число может быть представлено в виде произведения $M \cdot e^p$, где *M* – целое число, которое называется мантиссой, а *p* – целое число, называемое экспонентой или показателем степени. При этом *e* называется модулем экспоненты. В таком виде вещественные числа и представляются в электронных устройствах.



Вещественные числа одинарной и двойной точности



Кодировки

С числами все понятно. Но что насчет, например, символов? Правила, задающие однозначный переход от одного представления данных (как правило, числового) к другому (например, символьному), называется кодировкой. Для кодирования чисел наиболее известными являются ASCII и Unicode.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

0009B100   75 73 00 6B 69 6C 6C 73 65 72 76 65 72 00 73 63  us.killserver.sc
0009B110   72 69 70 74 55 73 61 67 65 00 73 74 72 69 6E 67  riptUsage.string
0009B120   55 73 61 67 65 00 73 61 79 00 74 65 6C 6C 00 00  Usage.say.tell..
0009B130   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0009B140   63 68 61 6C 6C 65 6E 67 65 52 65 73 70 6F 6E 73  challengeRespons
0009B150   65 20 25 69 20 25 69 00 63 68 61 6C 6C 65 6E 67  e %i %i.challeng
0009B160   65 52 65 73 70 6F 6E 73 65 20 25 69 00 63 6F 64  eResponse %i.cod
0009B170   75 6F 61 75 74 68 6F 72 69 7A 65 2E 61 63 74 69  uoauthorize.acti
0009B180   76 69 73 69 6F 6E 2E 63 6F 6D 00 52 65 73 6F 6C  vision.com.Resol
0009B190   76 69 6E 67 20 25 73 0A 00 43 6F 75 6C 64 6E 27  ving %s..Couldn'
0009B1A0   74 20 72 65 73 6F 6C 76 65 20 61 64 64 72 65 73  t resolve addres
0009B1B0   73 0A 00 00 00 00 00 00 00 00 00 00 00 00 00 00  s...............
0009B1C0   25 73 20 72 65 73 6F 6C 76 65 64 20 74 6F 20 25  %s resolved to %
0009B1D0   69 2E 25 69 2E 25 69 2E 25 69 3A 25 69 0A 00 61  i.%i.%i.%i:%i..a
0009B1E0   75 74 68 6F 72 69 7A 65 20 73 65 72 76 65 72 20  uthorize server
```

USASCII code chart

$b_7 b_6 b_5$ → / Bits / $b_4 b_3 b_2 b_1$ ↓ / Row ↓	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Column →	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

Понятие алгоритма

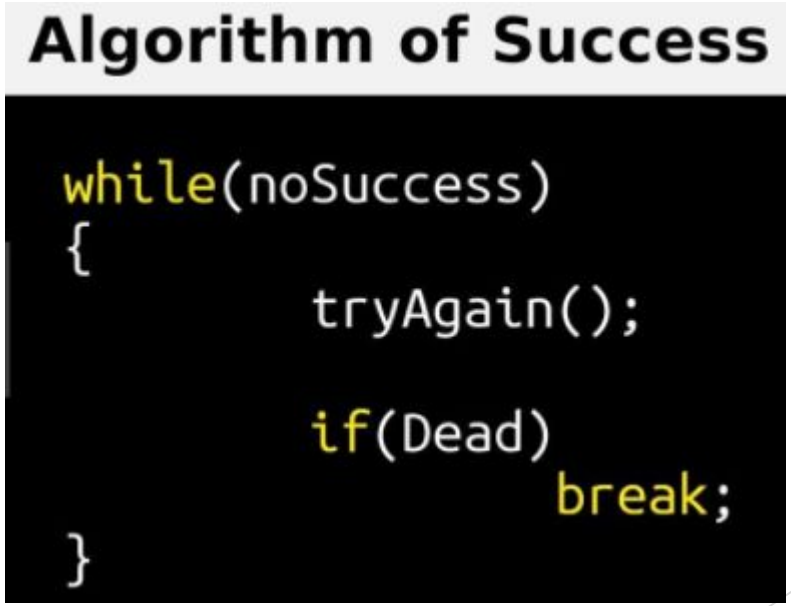
Алгоритм – это корректно определенная вычислительная процедура, представляющая собой конечною последовательность действий, понятная исполнителю.

Примеры алгоритмов:

- Рецепт
- Инструкция
- Программа

Свойства:

1) Корректность
2) Сложность
3) Требование к памяти
4) Способность к распараллеливанию



Основные элементы алгоритма

Алгоритм можно воспринимать как некоторую программу. Она содержит в себе основные элементы, такие как:

1) Линейная последовательность действий
2) Условия и условные переходы
3) Циклы