8 Short integers, integers, and ascii encoding

- A 16-bit number has values ranging from $0 \ldots 2^{16} - 1$ (65535). We call these the 'face values.'
- A *short int* is a 16-bit number interpreted to permit negative values.
- Those in the range $0 \ldots 32767$ retain their face values.
- Those in the range $32768 \ldots 65535$ represent *negative numbers* by subtracting 65536.
- If a 16-bit integer has face value x but encodes the value y, then $65536 - x$ encodes the value $-y$.
- This encoding is called *2s complement*.
- The C keyword `short` means 'short int' and is used to declare 16-bit integer variables.

  `short` variables are not used except where memory is scarce, like on rocket ships.

Example. Convert 1234, −1234, and 5678 to short int.
(1) To convert 1234 to short int, convert it to hex

```
1234 = 16 * 77 + 2  (division)
77 = 16 * 4 + 13
1234 = 16 * ( 16 * 4 + 13 ) + 2 =
4 * 16^2 + 13 * 16 + 2
Hex digits 4, 13, 2 ->  4d2 hex.
This is 3 hex digits or 12 bits.  We want 4 hex digits,
so pad it out.
ANSWER: 1234 = (04d2)_16
```

(2) To get −1234, subtract it from 65536: 64302. To get −1234, convert 64302 to hex.

```
64302 = 16 * 4018 + 14
4018  = 16 * 251 + 2
251 = 16 * 15 + 11
15 = 16 * 0 + 15   4 hex digits 15,11,2,14:  fb2e hex
```

(3) To get 5678, convert to hex.
Following the by now familiar procedure

```
5678 =    (((1*16 + 6) * 16 + 2 ) * 16 + 14
Hex digits 1,6,2,14 left to right: 162e
```

8.1 Practical reasons

The rule for adding short ints is: add from right to left in the usual way, but if there is a carry at the end, discard it.

Equivalently: addition of short integers is addition modulo 2^{16} (65536).

An integer x is *in short integer range* if $-32768 \leq x \leq 32767$.

(8.1) Lemma *If x, y, and $x + y$ are in short integer range, then addition modulo 65536 gives the correct answer.* ∎

For example, convert 5678 and -1234 to short integers, add as short integers, and convert the result back to decimal.

In other words, calculate

```
    1    6    2    e
+   f    b    2    e
        1
--------------------
    1    1    5    c
```

8.2 An easy way to compute the negative of a short int

This makes good sense in computer hardware. Bearing in mind that $65535 = (ffff)_{16} + 1$, to find the negative of a short int, subtract from `ffff` and add 1. This is easier because there is no 'borrowing.' So to calculate the short int encoding of -1234,

```
1234 = (04d2)_16

                 f    f    f    f
            -    0    4    d    2
                 -----------------
                 f    b    2    d
            +                   1
                 -----------------
                 f    b    2    e
```

8.3 32-bit integers

These are the most commonly used. Their range is about ±2bn.

- The precise range is from -2^{31} to $2^{31} - 1$.
- 2s complement encoding is used.
- Addition is modulo 2^{32}. Lemma 8.1 applies.

8.4 Ascii code

The ASCII code is the generally accepted way to represent printable characters (and some non-printable, such as control characters) in memory, economically.

These codes fit into bytes, that is, 8-bit binary numbers. Therefore they go from 0 to 255 decimal, 0 to 377 octal, and 0 to ff hex. On second thoughts, only the codes with high-order bit 0 are shown. The table shows codes from $(20)_{16} = 32$ (space) to $(7f)_{16} = 127$ (DEL).

Modern 'unicode' is much more complicated.

For example:

```
"Good morning, madam," to Eve said Adam,
is represented as
 22 47 6f 6f 64 20 6d 6f 72 6e 69 6e 67 2c 20 6d 61 64 61 6d 2c
 22 20 74 6f 20 45 76 65 20 73 61 69 64 20 41 64 61 6d 2c 0a 00
```

That 00: more about that later. A table of the ascii codes is given below, split over two pages. (It gives everything from 32 onwards; from 0 to 31 decimal, the characters are control characters, etcetera, not printable).

octal	dec	hex		octal	dec	hex	
040	32	20	SPACE	120	80	50	P
041	33	21	!	121	81	51	Q
042	34	22	"	122	82	52	R
043	35	23	#	123	83	53	S
044	36	24	$	124	84	54	T
045	37	25	%	125	85	55	U
046	38	26	&	126	86	56	V
047	39	27	'	130	88	58	X
050	40	28	(	131	89	59	Y
051	41	29	)	132	90	5A	Z

052	42	2A	*	133	91	5B	[
053	43	2B	+	134	92	5C	\ '\\'
055	45	2D	-	135	93	5D	]
056	46	2E	.	136	94	5E	^
057	47	2F	/	137	95	5F	_
060	48	30	0	140	96	60	`
061	49	31	1	141	97	61	a
062	50	32	2	142	98	62	b
063	51	33	3	143	99	63	c
064	52	34	4	144	100	64	d
065	53	35	5	145	101	65	e
066	54	36	6	146	102	66	f
067	55	37	7	147	103	67	g
070	56	38	8	150	104	68	h
071	57	39	9	151	105	69	i
072	58	3A	:	152	106	6A	j
073	59	3B	;	153	107	6B	k
074	60	3C	<	155	109	6D	m
075	61	3D	=	156	110	6E	n
076	62	3E	>	157	111	6F	o
077	63	3F	?	160	112	70	p
100	64	40	@	161	113	71	q
101	65	41	A	162	114	72	r
103	67	43	C	163	115	73	s
104	68	44	D	164	116	74	t
105	69	45	E	165	117	75	u
106	70	46	F	166	118	76	v
107	71	47	G	167	119	77	w
110	72	48	H	170	120	78	x
111	73	49	I	171	121	79	y
112	74	4A	J	172	122	7A	z
113	75	4B	K	173	123	7B	{
114	76	4C	L	174	124	7C	|
115	77	4D	M	175	125	7D	}
116	78	4E	N	176	126	7E	~
117	79	4F	O	177	127	7F	DEL