CSC 143

Two' s complement

10's complement

- How to represent negative numbers?
  - Use a sign → but −0 is the same as +0
  - 10's complement
- Example
  pick a number of digits (sign + magnitude):
  e.g. 3
  positive numbers: +18 → 018
  negative numbers: −18 → 982
- Why?

−18 in 10's complement

- Start with the positive number (3 digits)
  018
- Write the 9's complement (0→9, 1→8, etc.)
  981
- To get the 10's complement, add 1
  982
- Same as doing 1000 – 18
- With 3 digits,
  900 to 999 is -100 to -1
  000 to 099 is 0 to 99

What about 0?

- Using 3 digits
- +0 is 000
- -0?
  000 (+0)
  999 (9's complement)
  999 + 1 = 000 + carry = 1(10's complement)
  Always ignore the carry, so
  -0 is 000
- Same as +0

Rules

- Positive numbers start with 0
- Negative numbers start with 9
- Apply the usual rules of arithmetic
  e.g. with 3 digits:
  - 25 – 30 = 025 + 970 = 995
    995 is the usual -5 (995 = 1000 – 5)
  - -10 -15 = 990 + 985 = 2975 (ignore the carry!). 975 is the usual -25
- Overflow if the carry going into the sign digit (0 or 9) is not equal to the carry coming out of it. Remedy → use more digits.

2's complement

- Positive numbers start with 0
  e.g with 4 digits
  7 is 4 + 2 + 1 = 111_2 = 0111_2
- Negative numbers?
  Start with the positive value: +7 -> 0111_2
  1's complement: 1000
  2's complement (add 1): 1001 (= -8 + 1)
- Positive numbers start with 0, negative numbers start with 1

Examples

- Rules: 1 + 1 = 10 (0, carry =1)

  0 + 1 = 1, 1 + 0 = 1, 0 + 0 = 0

  Ignore any carry out of the sign bit (overflow?)
- 37 + 19 (with 8 digits) = 56

37 =	0	0	1	0	0	1	0	1
19 =	0	0	0	1	0	0	1	1
56 =	0	0	1	1	1	0	0	0

- -37 + 19 (with 8 digits) = -18

-37 =	1	1	0	1	1	0	1	1
19 =	0	0	0	1	0	0	1	1
-18=	1	1	1	0	1	1	1	0

Overflow

- 100 + 50 (with 8 digits) = 150 (too big!)

100 =	0	1	1	0	0	1	0	0
50 =	0	0	1	1	0	0	1	0
overflow!	1	0	0	1	0	1	1	0

The carry into the sign bit is +1 ≠ carry out of the sign bit is 0

- Fix: use 9 digits

100 =	0	0	1	1	0	0	1	0	0
50 =	0	0	0	1	1	0	0	1	0
150 =	0	1	0	0	1	0	1	1	0

What about multiplication?

- Works as usual if the number of digits is enough to accommodate the answer
- **With 3 digits: -3 = 101, -4 = 100**
- **However, -4 x -3 = 12 = 01100_2 doesn't fit into 3 digits**
- **Fix: sign extend -3 and -4 to as many bits as necessary**
- **How many? Safe approach: here, double the number of digits. In our example, switch to 6 digits**
- **-4 = 111100, -3 = 111101**
- **Do the multiplication and keep only the last 6 digits for all operations = 001100**

Rules (1)

- For any number, can always remove all 0's or all 1's on the left except the last 0 and 1:

```
000110 is 0110 (= +6)
111011 is 1011 (= -5)
```

- To convert to decimal form
  - If positive: 01101 = $2^3 + 2^2 + 2^0 = 13$
  - If negative: 10111
    $-2^4 + 2^2 + 2^1 + 2^0 = -9$
    or
    convert to positive:
    1's complement = 01000
    2's complement = 01000 + 1 = 01001
    and 01001 = $2^3 + 2^0 = 9$
    so 10111 = -9

Rules (2)

- Addition: a + b
  - Pad a and b on the left to max(number of digits of a, number of digits of b) + 1 → never any overflow, always ignore any outgoing carry on the leftmost digits.
  - -5 = 1011 (4 digits)
    -3 = 101 (3 digits)
    pad to max(4,3) + 1 = 5
    -5 + -3 = 11011 + 11101

```
  ¹1 ¹1 ¹0 ¹1  1
+  1  1  1  0  1
---------------
   1  1  0  0  0
```

(the outgoing carry = 1 on the leftmost digits is ignored)

11000 = 1000 = -8

Rules (3)

- Multiplication: a × b
  - Pad a and b on the left to m = number of digits of a + number digits of b: never any overflow, always ignore any outgoing carry on the leftmost digits, for all steps work with just m digits

```
-5 = 1011 (4 digits)
-3 = 101  (3 digits)

work with 4 + 3 = 7 digits
for all operations

-5 × -3 = 0001111 = 01111 = 15
```

```
   1 1 1 1 0 1 1
× 1 1 1 1 1 0 1
---------------
  ¹1 ¹1 ¹1 1 0 1 1
+ 1 1 0 1 1 . . .
---------------
  1 1 0 0 1 1 1
+ 1 0 1 1 . . .
---------------
 ¹0 ¹1 1 1 1 1 1
+ 0 1 1 . . . .
---------------
  ¹1 1 0 1 1 1 1
+ 1 1 . . . . .
---------------
  1 0 0 1 1 1 1
+ 1 . . . . . .
---------------
  0 0 0 1 1 1 1
```