CS1800
9/19 - Fri ☺

Admin

- HW1 due 9/23 9pm
- in two weeks, ... Quiz!!
  - ↳ 10/3 in class

Agenda

1. Rep of numbers
2. unsigned values, bases
3. signed values, arithmetic

0. Review — domain = integers

(a) $\forall x \exists y (x + y = 0)$ For all x, there is a y s.t. $x+y=0$ True!

(b) $\exists y \forall x (x + y = x)$ There exists y, s.t. for all x, $x+y=x$ True!

(c) $\exists x \forall y (x + y = x)$ there exists x, s.t. for all y $x+y=x$ False

1. Rep of numbers

↳ George Boole ~1800s — propositional logic T/F and/or/not

Claude Shannon ~1900s - generalize logic yes/no
cat/dog
1/0
on/off

 decimal representation (base 10)

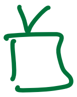
$$135_{10} = 5\cdot1 + 3\cdot10 + 1\cdot100$$
$$= 5\cdot10^0 + 3\cdot10^1 + 1\cdot10^2$$

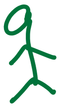
decimal counting (10 digits 0-9)

0
1
2
...
8
9
10
11
...

Same in all bases!

what changes?
- exponents (rep)
- digits

 binary (base 2)

 base 10
base 2
base 8
base 16

decimal addition

$$\begin{array}{r} 3 \\ +2 \\ \hline 5 \end{array} \qquad \begin{array}{r} \overset{1}{7} \\ +8 \\ \hline 15 \end{array}$$

carry

binary representation (convert from base 2 to base 10)

$$10110_2 = 0\cdot2^0 + 1\cdot2^1 + 1\cdot2^2 + 0\cdot2^3 + 1\cdot2^4$$
$$= 0 + 2 + 4 + 0 + 16$$
$$= 22_{10}$$

binary counting (digits 0, 1) bit = "binary digit"

0	0
1	1
10	2

 to  (base 10 to base 2)

- division method

11	3
100	4
101	5
110	6
111	7

- divide by 2
- keep the remainder
- divide quotient by 2

ex $135_{10} = \underline{\ ?\ }_2$

$135 \div 2 = 67 \text{ R } 1$

$67 \div 2 = 33 \text{ R } 1$

$33 \div 2 = 16 \text{ R } 1$

$16 \div 2 = 8 \text{ R } 0$

$8 \div 2 = 4 \text{ R } 0$

$4 \div 2 = 2 \text{ R } 0$

$2 \div 2 = 1 \text{ R } 0$

$1 \div 2 = 0 \text{ R } 1$

flip!

10000111_2

2. unsigned values, bases

- unsigned: not specify pos/neg (all positive)

- base 10 (0-9)
  base 2 (0, 1)
  base 8 (0-7)
  base 16 (0-9 A B C D E F)
  10 11 12 13 14 15

conversion / counting same in any base b

$37_{10} = \underline{\ ?\ }$

convert 37_{10} to base 2

$37 \div 2 = 18 \text{ R } 1$

$18 \div 2 = 9 \text{ R } 0$

$9 \div 2 = 4 \text{ R } \underline{1}$

$4 \div 2 = 2 \text{ R } 0$

$2 \div 2 = 1 \text{ R } 0$

$1 \div 2 = 0 \text{ R } 1$

100101_2

convert 37_{10} to base 8

$37 \div 8 = 4 \text{ R } 5$

$4 \div 8 = 0 \text{ R } 4$

45_8

sanity check: do we get back the same?

$45_8 = \underline{\quad}_{10}$?

$5 \cdot 8^0 + 4 \cdot 8^1 = 5 \cdot 1 + 4 \cdot 8 = 5 + 32 = 37_{10}$!!

$100101_2 = \underline{\quad}_{10}$?

$1 \cdot 2^0 + 1 \cdot 2^2 + 2^5$

$= 1 + 4 + 32$

$= 37_{10}$ 🙂

convert base x to base y
where $x \neq y \neq 10$?

- general: convert to base 10 in between
- fun shortcut: if powers of 2, split up into chunks – lookup table

ex from base 2 to base 16
split the base 2 # into chunks of 4 bits each
convert or look up the 4 bits at a time

Decimal	Hex	4-bit Binary	Decimal	Hex	4-bit Binary
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	10	A	1010
3	3	0011	11	B	1011
4	4	0100	12	C	1100
5	5	0101	13	D	1101
6	6	0110	14	E	1110
7	7	0111	15	F	1111

convert 1011100_2 to base 16

make it 8 bits long (pad)

$\underbrace{0101}_{5} \quad \underbrace{1100}_{C}$

$\boxed{5C_{16}}$

3. Signed values, arithmetic

↳ still unsigned, doing math

- so far, very 
- turning number rep. into real life

ex addition ↗carry↘

$$\begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \qquad \begin{array}{r} {}^{1}1 \\ +1 \\ \hline 10 \end{array} \qquad \begin{array}{r} {}^{1}1 \\ 1 \\ +1 \\ \hline 11 \end{array} \qquad \begin{array}{r} 10 \\ +1 \\ \hline 11 \end{array} \qquad \begin{array}{r} {}^{1}10 \\ +10 \\ \hline 100 \end{array}$$

practice

$$\begin{array}{r} {}^{1}10 \\ +11 \\ \hline 101 \end{array} \qquad \begin{array}{r} {}^{1}1{}^{1}01 \\ +101 \\ \hline 1010 \end{array} \qquad \begin{array}{r} {}^{1}1{}^{1}1{}^{1}1 \\ +111 \\ \hline 1110 \end{array}$$


notice!

- sometimes, we needed an extra bit
- everything positive so far

need 

- pos/neg values
- # bits per value
  - separate values
  - physical limitations

10110 is two numbers?

Actual  number system:

- # bits for every number
- pos, neg values
- addition etc.

two's complement

↳ k-bit two's complement
always k bits for every number

(IRL k = 32, 64
for us k = 3, 4, 8, ...)

pos, neg values

- leftmost bit = sign bit
  0 pos, 1 neg


sign + magnitude +38 ✗✗ 🙁
-38 ✗

addition

```
 ˈ
 110      -2
+111      -1
----
1101
‾
```

throw away!

101 = -3 🙂

conversion

- positive: just convert!
  pad if necessary
- negative: take complement
  add one

Ⓔⓧ 3 bit two's complement

counting (non-neg)		counting (neg)	
000	0	100	-4
001	1	101	-3
010	2	110	-2
011	3	111	-1

sign sign

```
 001
+010
----
 011   = 3 🙂
```

```
 ˈ
 010     2
+110    -2
----
1000    = 0 🙂
‾
```

throw away!

Ⓔⓧ 4 bit two's complement

3_{10} = 11
= 0011 (pad)

-3_{10} = 1100 (complement)
+1 (add one)

1101

7_{10} = 111
= 0111 (pad)

-7_{10} = 1000 (complement)
+1 (add one)

1001