More about datatypes...

"Closure" properties for integers + floating point:

$$\text{int} \; \boxed{?} \; \text{int} \longrightarrow \text{int}$$

($\boxed{?}$ could be +, -, *, /, %, ...)

$$\text{float} \; \boxed{?} \; \text{float} \longrightarrow \text{float}$$

$$\text{int} \; \boxed{?} \; \text{float} \longrightarrow \text{float}$$

Note: there is an extended "PEMDAS" for C++ operators. See "precedence of operators in C++".

(Note: no built-in exponentiation!

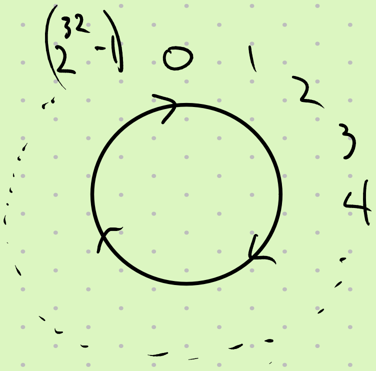
$x\text{^}y \neq x^y$ (it's XOR))

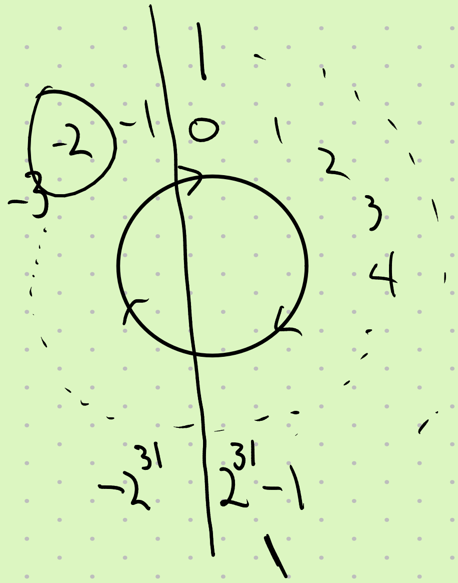
Remark on integer overflow: say we have this:

<u>unsigned</u> int x,y; // no negatives: $0 \leq x,y < 2^{32}$

what if $x+y \geq 2^{32}$?

e.g. $(2^{32}-1)+3=2$





e.g. $(2^{31}-1)+(2^{31}-1)$

$$= -2$$

$$(2^{31}-1)+(2^{31}-1)$$

$$= 2\cdot 2^{31} - 2$$

$$= \cancel{2^{32}} - 2$$

look up "modular arithmetic"

Booleans

bool b; // b can be true or false

examples of boolean expressions:

(say x,y integers & b,c booleans)

(x == 7)

(x != y)

(x > y)

(x <= y)

(b && c) // true only if b and c are true

```
(b || c)  // true if either b or c
                          is true
 (!b)  // true ⇔ b is false
```

E.g. $(x < y)$ is the same thing as

$!(x >= y)$

```
if(!b) { ... }
```

= vs == foot-gun:

```
int x,y;
  ⋮
if (x = y) { ... }
```

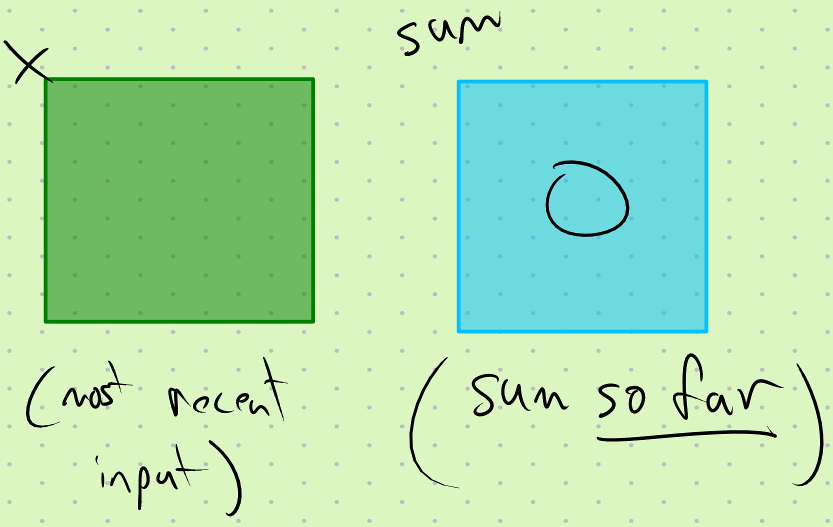
compiler error? Nope!

⊛ compile via g++ -Wall (or use my Makefiles!)

Exercise: read integers & computes the sum of all of them.

Note/hint: can do this with only 2 variables.

Ideas?


x
sum
0
(most recent input)
(sum so far)