Proofs are programs

$P \wedge Q \Rightarrow P$

$P \wedge Q \Rightarrow P$
$P \wedge Q \Rightarrow Q$

Constructive

$P \wedge Q$??

(proof of P, proof of Q)

$P \wedge Q$ like `'p * 'q`
$P \Rightarrow Q$ like `'p -> 'q`

```
let f (x, y) : 'p * 'q : 'p = x
f : 'p * 'q -> 'p
```

f: $P \wedge Q \Rightarrow P$

$P \vee Q$ like variant

```
type 'p 'q or = P of 'p | Q of 'q
```

$$((P \vee Q) \wedge (P \Rightarrow S) \wedge (Q \Rightarrow S)) \Rightarrow S$$

```
let g (parq : 'p 'q or) (ps : 'p -> 's) (qs : 'q -> 's) : 's
  = match parq with
     | P p -> ps p
     | Q q -> qs q

g : 'p 'q or -> ('p -> 's) -> ('q -> 's) -> 's
```

Practical utility

- Manipulating data   separate reasoning w/ propositions?

No!

- Write program and reason about it
- Google "natural number game"

- succ (a : mynat) : mynat ← data
- add_zero (a : mynat) : a + 0 = a ← proposition

↳ functions applied to natural numbers