Let's see if you can use the tactics we've learnt to prove $x+1=y+1 \implies x=y$. Try this one by yourself; if you need help then click on "Show more help!".

Level completed! 🎉

Here's a completely backwards proof:

```
intro h
apply succ_i
repeat rw [s
exact h
```

Objects:

y x : ℕ

Assumptions:

h : `succ x = y + 1`

Goal:

`x = y`

```
rw[<-succ_eq_add_one y]
at h
```

Retry

Active Goal

Objects:

y x : ℕ

Assumptions:

h : `succ x = succ y`

Goal:

`x = y`

```
apply succ_inj at h
```

Retry

Active Goal

Objects:

y x : ℕ

Assumptions:

h : `x = y`

Goal:

`x = y`

```
exact h
```

Retry

`level completed!` 🎉

succ_inj

`(a b : ℕ) (h : MyNat.succ a = MyNat.succ b) : a = b`

Statement

If a and b are numbers, then `succ_inj a b` is the proof that $(\text{succ}(a)=\text{succ}(b)) \implies a=b$.

More technical details

There are other ways to think about `succ_inj`.

You can think about `succ_inj` itself as a function which takes two numbers a and b as input, and outputs a proof of $(\text{succ}(a)=\text{succ}(b)) \implies$