is an axiom of Lean that recursion is a valid way to define functions from types such as the naturals.

Let's define a new function `pred` from the naturals to the naturals, which attempts to subtract 1 from the input. The definition is this:

```
pred 0 := 37
pred (succ n
```

We cannot subtract one from 0, so we just return a junk value. As well as this

Next

Objects:

a **b** : ℕ

Assumptions:

h : `succ a = succ b`

Goal:

`pred (succ a) = b`

`rw[h]` Retry

Active Goal

Objects:

a **b** : ℕ

Assumptions:

h : `succ a = succ b`

Goal:

`pred (succ b) = b`

`rw[pred_succ b]` Retry

Active Goal

Objects:

a **b** : ℕ

Assumptions:

h : `succ a = succ b`

Goal:

`b = b`

`rfl` Retry

level completed! 🎉

pred_succ

`(n : ℕ) : MyNat.pred (MyNat.succ n) = n`

`pred_succ n` is a proof of `pred (succ n) = n`.