

# 16 - Writing Our Own Lemma

Lean: First Steps

Tariq Rashid

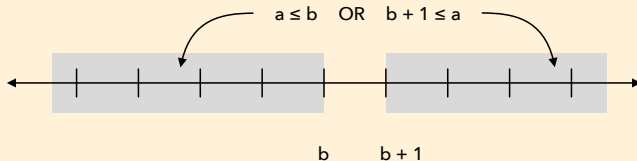
November 17, 2024

# Writing Our Own Lemmas

- Mathlib has many lemmas and theorems for us to use.
- We can also write our own.
- We'll create a small convenient lemma about the natural numbers.

# Lemma

- If we pick a natural number  $b$ , then any other natural number  $a$  must be less than or equal to  $b$ , or greater than or equal to  $b + 1$ .



- Let's create a lemma for natural numbers  $a$  and  $b$

$$a \leq b \vee b + 1 \leq a$$

- Example: setting  $b = 7$  lemma says  $a \leq 7$  or  $8 \leq a$ .

---

*-- 16 - Writing And Using Our Own Lemma*

```
import Mathlib.Tactic
```

```
lemma Nat.le_or_succ_le (a b:  $\mathbb{N}$ ):  $a \leq b \vee b + 1 \leq a$  := by  
  rw [Nat.le_or_succ_le]  
  exact le_or_lt a b
```

```
example {c :  $\mathbb{N}$ } :  $c \leq 2 \vee 3 \leq c$  := by  
  exact Nat.le_or_succ_le c 2
```

---

- `lemma Nat.le_or_succ_le` tells Lean we want to create a new lemma named `Nat.le_or_succ_le`.
  - `Nat.` prefix is conventional for lemmas about natural numbers.
  - `le_or_succ_le` tries to follow the naming convention to describe what the lemma is about.
- `a` and `b` declared as natural numbers, and states the lemma's proposal  `$a \leq b \vee b + 1 \leq a$` .
- The round brackets `(a b : ℕ)` require anyone using the lemma to always provide `a` and `b` as parameters.

- The first line of the proof rewrites the goal using Mathlib lemma `Nat.add_one_le_iff`. Let's see its definition:

```
theorem add_one_le_iff : n + 1 ≤ m ↔ n < m :=
```

- The `rw` tactic is like “find and replace”, so it looks at the goal `a ≤ b ∨ b + 1 ≤ a` and finds `b + 1 ≤ a` matches the LHS of the `Nat.add_one_le_iff` lemma. The matched `b + 1 ≤ a` is rewritten with the RHS of the lemma `b < a`.
- We'll see in the Infoview, the goal is now `a ≤ b ∨ b < a`.

- The final line of the proof applies Mathlib lemma `le_or_lt`. Let's look at its definition:

```
lemma le_or_lt (a b :  $\alpha$ ) :  $a \leq b \vee b < a :=$ 
```

- This matches the current goal exactly, so we can use `exact` to resolve the goal and complete the proof.
- Notice the definition of the Mathlib lemma `le_or_lt (a b :  $\alpha$ )` uses round brackets requiring us to provide `a` and `b` when we use it.

- Placing the cursor before `rw [Nat.add_one_le_iff]` shows the original proof goal.

---

$$\vdash a \leq b \vee b + 1 \leq a$$

---

- Moving the cursor to the start of the next line shows the rewritten goal.

---

$$\vdash a \leq b \vee b < a$$

---

- This proof goal now matches exactly the Mathlib lemma `le_or_lt a b`, and can be resolved with **exact**.



# Minimal Example

- When writing your own lemmas, it is considerate to provide a **minimal example** showing how to use them.
- The provided example illustrates how to prove  $c \leq 2 \vee 3 \leq c$  using our lemma.
- With `a` set to `c`, and `b` set to `2`, the lemma becomes  $c \leq 2 \vee 3 \leq c$ .
  - This matches the proof goal exactly, allowing us to use **exact** with our lemma to complete the proof.

# Easy Exercise

- Write a lemma for **integers**  $a$  and  $b$  that says

$$a \leq b \vee b + 1 \leq a$$

- You can copy the provided lemma for natural numbers and modify it to work with integers.
- Create a minimal example illustrating how your lemma can prove the following for any integer  $c$ .

$$c \leq -5 \vee -4 \leq c$$