# Automating Natural Deduction 1

Simon Foster Jim Woodcock
University of York

18th August 2022

# Overview

1. Low-Level Proof Scripts
2. Natural Deduction Rules for Propositional Calculus
3. Automation with the Classical Reasoner

# Outline

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Motivation

- The simp tactic uses equations ($s = t$) to rewrite and simplify a goal.
- Proof by simplification is the gold standard for efficiency and usability.
- But it's not always possible to prove a goal with just the simplifier.
- Isabelle provides tactics for automating natural deduction.
- One of the core reasoning techniques, alongside the simplifier.
- You need to understand the use of low-level proof scripts.

**Proof Script**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  apply assumption
  apply (rule disjI1)
  apply assumption
  done
```

# Proof Scripts and Tactics

- Low-level proof language: like the "machine code" for proofs
- Proof script: sequence of commands acting on a proof state
- Proof state: subgoals
- Subgoals: collection of hypotheses and outstanding conclusions
- Commands invoke proof tactics to decompose and eliminate the goals

- Proof script uses the keyword to continue the proof
- Tactics can fail if the subgoal is in the wrong form
- Proof is complete once all subgoals are eliminated
- Script terminated with (or )

# Proof Scripts and Tactics

- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.

- Proof script uses the keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with (or ).

# Proof Scripts and Tactics

- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.

- Proof script uses the keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with (cf. ).

# Proof Scripts and Tactics

- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.

# Proof Scripts and Tactics

- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.

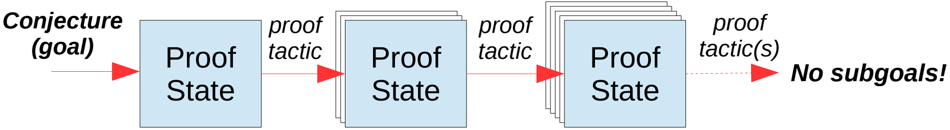
- Proof script uses the keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with (cf. ).

# Proof Scripts and Tactics

- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.
- Proof script uses the keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with (cf. ).

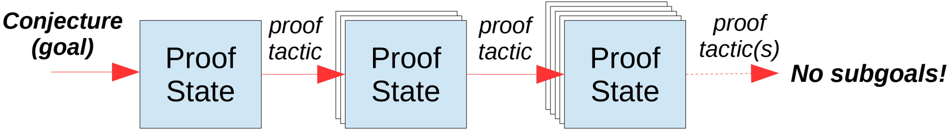
# Proof Scripts and Tactics

- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.



# Proof Scripts and Tactics

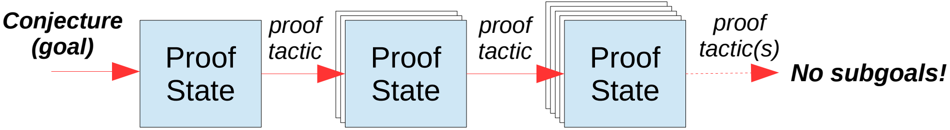
- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.




- Proof script uses the `apply` keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with `done` (cf. `qed`).

# Proof Scripts and Tactics

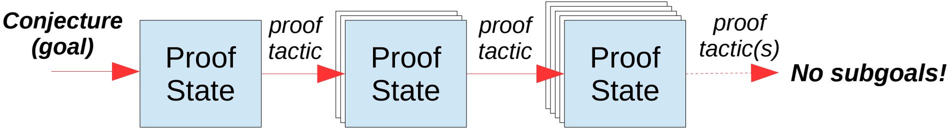
- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.




- Proof script uses the `apply` keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with `done` (cf. `qed`).

# Proof Scripts and Tactics

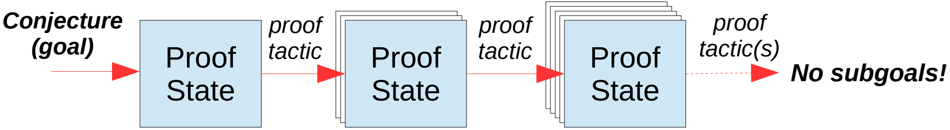
- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.




- Proof script uses the `apply` keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with `done` (cf. `qed`).

# Proof Scripts and Tactics

- Low-level proof language are the “machine code” for Isar.
- Proof script: sequence of commands acting on a proof state.
- Proof state: subgoals.
- Subgoals: collection of hypotheses and outstanding conjectures.
- Commands invoke proof tactics to decompose and refine the goals.




- Proof script uses the `apply` keyword to execute a tactic.
- Tactics can fail if the subgoal is in the wrong form.
- Proof is complete once all subgoals are eliminated.
- Script terminated with `done` (cf. `qed`).

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \ldots; P_n]\!] \Longrightarrow Q$.
- $P_1 \ldots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal, the thing to be proved.
- `assumes P1 ... Pn shows Q` creates a single initial subgoal:
  - $[\![P_1; \ldots; P_n]\!] \Longrightarrow Q$
- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
  - Split into several additional subgoals by introduction or elimination.
  - Manipulate the assumptions or insert further ones, e.g. from theorems.
  - Discharge trivial subgoals, e.g. True or $P \Longrightarrow P$ using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots ; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- `assumes P1 ... Pn shows Q` creates a single initial subgoal:

  $$[\![P_1; \cdots ; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- `assumes P1 ... Pn shows Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P ⟹ P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots ; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- `assumes P1 ... Pn shows Q` creates a single initial subgoal:

  $$[\![P_1; \cdots ; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P ⟹ P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- `assumes P1 ... Pn shows Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P ⟹ P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1;\ \cdots\ ;\ P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- `assumes P1 ... Pn shows Q` creates a single initial subgoal:

  $$[\![P_1;\ \cdots\ ;\ P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- **`assumes`** `P1 ... Pn` **`shows`** `Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- **assumes** `P1 ... Pn` **shows** `Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots ; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- **assumes** `P1 ... Pn` **shows** `Q` creates a single initial subgoal:

  $$[\![P_1; \cdots ; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- **`assumes`** `P1 ... Pn` **`shows`** `Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- **assumes** `P1 ... Pn` **shows** `Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$
- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- `assumes P1 ... Pn shows Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Proof State

- The proof state consists of $n$ subgoals.
- The aim is to discharge (remove by proof) all remaining subgoals.
- Subgoals have the form $[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$.
- $P_1 \cdots P_n$ is a set of hypotheses – can be used to prove the subgoal.
- $Q$ is the conclusion of the subgoal; the thing to be proved.
- **assumes** `P1 ... Pn` **shows** `Q` creates a single initial subgoal:

  $$[\![P_1; \cdots; P_n]\!] \Longrightarrow Q$$

- Tactics often act on the first subgoal, but some act on all (`simp_all`).
- We can manipulate a subgoal in several ways:
- Split into several additional subgoals by introduction or elimination.
- Manipulate the assumptions or insert further ones, e.g. from theorems.
- Discharge trivial subgoals, e.g. `True` or `P` $\Longrightarrow$ `P` using `assumption`.

# Outline

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning.
2. Elimination rules (erule tactic): forwards reasoning.

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning
2. Elimination rules (erule tactic): forwards reasoning

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning.
2. Elimination rules (erule tactic): forwards reasoning.

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning
2. Elimination rules (erule tactic): forwards reasoning

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning.
2. Elimination rules (erule tactic): forwards reasoning.

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning.
2. Elimination rules (erule tactic): forwards reasoning.

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning.
2. Elimination rules (erule tactic): forwards reasoning.

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning.
2. Elimination rules (erule tactic): forwards reasoning.

# Natural Deduction

## Natural Deduction Rules

$$\frac{\Gamma \vdash P_1 \quad \Gamma \vdash P_2 \cdots \Gamma \vdash P_n}{\Gamma \vdash Q}$$

- Meaning: a proof of the conclusion $Q$ follows from premises $P_1 \cdots P_n$.
- $\Gamma \vdash Q$ is a sequent: valid when $Q$ holds subject to the hypotheses in $\Gamma$.

## Example (Syllogism)

$$\frac{\text{Socrates is a man} \qquad \text{All men are mortal}}{\text{Socrates is a mortal}}$$

Two main kinds of deduction rules:

1. Introduction rules (rule tactic): backwards reasoning.
2. Elimination rules (erule tactic): forwards reasoning.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

Introduction Rules

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

Introduction Rules

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

Introduction Rules

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

### Introduction Rules

conjI TrueI

disjI1 disjI2

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

**Introduction Rules**

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \land Q}\ \texttt{conjI} \qquad \frac{-}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \lor Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \lor Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

**Introduction Rules**

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ \texttt{conjI} \qquad \frac{}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

**Introduction Rules**

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ \texttt{conjI} \qquad \frac{-}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

**Introduction Rules**

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ \texttt{conjI} \qquad \frac{-}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

**Introduction Rules**

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ \texttt{conjI} \qquad \frac{-}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

**Introduction Rules**

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ \texttt{conjI} \qquad \frac{-}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

## Introduction Rules

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ \texttt{conjI} \qquad \frac{-}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Introduction Rules

- Question: how do we prove the conclusion $P$?
- By finding subgoals $P_1 \cdots P_n$, based on current subgoal's conclusion.
- Proof by introduction is often called backwards reasoning.

**Introduction Rules**

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ \texttt{conjI} \qquad \frac{-}{\Gamma \vdash \mathit{True}}\ \texttt{TrueI}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ \texttt{disjI1} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ \texttt{disjI2}$$

- Invoke introduction rules using the rule tactic, e.g. `apply(rule conjI)`.
- Matches subgoal's conclusion, copies any hypotheses to new subgoals.
- Most rules are safe introduction rules, but `disjI1` and `disjI2` aren't.

# Example: Proof by Introduction

## Natural Deduction Proof

asm

asm disjI1

conjI

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \vee Q}\,\texttt{disjI1}}{P \vdash P \wedge (P \vee Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\text{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\text{asm}}{P \vdash P \lor Q}\,\text{disjI1}}{P \vdash P \land (P \lor Q)}\,\text{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \vee Q}\,\texttt{disjI1}}{P \vdash P \wedge (P \vee Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{P \vdash P \qquad P \vdash P \vee Q}{P \vdash P \wedge (P \vee Q)}\ \texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \lor Q}\,\texttt{disjI1}}{P \vdash P \land (P \lor Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \lor Q}\,\texttt{disjI1}}{P \vdash P \land (P \lor Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{-}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{\overline{\phantom{x}}\ \texttt{asm}}{P \vdash P}}{P \vdash P \vee Q}\,\texttt{disjI1}}{P \vdash P \wedge (P \vee Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \lor Q}\,\texttt{disjI1}}{P \vdash P \land (P \lor Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \lor Q}\,\texttt{disjI1}}{P \vdash P \land (P \lor Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \vee Q}\,\texttt{disjI1}}{P \vdash P \wedge (P \vee Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{}{P \vdash P}\,\texttt{asm}}{P \vdash P \lor Q}\,\texttt{disjI1}}{P \vdash P \land (P \lor Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{-}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{-}{P \vdash P}\,\texttt{asm}}{P \vdash P \lor Q}\,\texttt{disjI1}}{P \vdash P \land (P \lor Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction

## Natural Deduction Proof

$$\dfrac{\dfrac{-}{P \vdash P}\,\texttt{asm} \qquad \dfrac{\dfrac{-}{P \vdash P}\,\texttt{asm}}{P \vdash P \vee Q}\,\texttt{disjI1}}{P \vdash P \wedge (P \vee Q)}\,\texttt{conjI}$$

## Proof Script

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

## Proof State

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

**Proof State**

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

**Proof State**

1. $P \Longrightarrow (P \wedge (P \vee Q))$.

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

## Proof State

1. $P \Longrightarrow P$.
2. $P \Longrightarrow P \vee Q$.

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

## Proof State

1. $P \Longrightarrow P$.
2. $P \Longrightarrow P \vee Q$.

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

**Proof State**

1. $P \Longrightarrow P \vee Q.$

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

**Proof State**

1. $P \Longrightarrow P \vee Q$.

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

## Proof State

1. $P \Longrightarrow P.$

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

**Proof State**

No subgoals!

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# Example: Proof by Introduction (Again)

## Proof State

```
lemma "P ⟹ (P ∧ (P ∨ Q))"
  apply (rule conjI)
  (* Subgoal 1 *)
  apply assumption
  (* Subgoal 2 *)
  apply (rule disjI1)
  apply assumption
  done
```

# More Introduction Rules

### Introduction Rules

impI conjI refl

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

**Introduction Rules**

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)   (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

**Introduction Rules**

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)   (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)   (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

## Introduction Rules

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# More Introduction Rules

**Introduction Rules**

$$\frac{P, \Gamma \vdash Q}{\Gamma \vdash P \longrightarrow Q}\ \texttt{impI} \qquad \frac{\Gamma \vdash P \longrightarrow Q \qquad \Gamma \vdash Q \longrightarrow P}{\Gamma \vdash P \longleftrightarrow Q}\ \texttt{iffI} \qquad \frac{-}{\Gamma \vdash t = t}\ \texttt{refl}$$

```
lemma "P ⟶ (P ∧ (True = True))"
apply (rule impI)  (* P ⟹ (P ∧ (True = True)) *)
apply (rule conjI) (* P ⟹ P and P ⟹ (True = True)) *)
apply (assumption) (* P ⟹ (True = True)) *)
apply (rule refl)
done
```

# Elimination Rules

- What can we deduce from hypothesis $P$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

- Elimination rules use erule tactical (`erule conjE`)
- Find first hypothesis matching first premise of elimination rule
- Replace subgoal with remaining premises: $[\![P; Q]\!] \Longrightarrow R$

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical `apply (erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical **`apply`** `(erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical **`apply`** `(erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Elimination Rules

- What can we deduce from hypothesis $P_i$? Refine a subgoal.

## Elimination Rules for Propositional Calculus

$$\frac{P, Q, \Gamma \vdash R}{P \wedge Q, \Gamma \vdash R}\ \texttt{conjE} \qquad \frac{P, \Gamma \vdash R \qquad Q, \Gamma \vdash R}{P \vee Q, \Gamma \vdash R}\ \texttt{disjE}$$

$$\frac{\Gamma \vdash P \qquad Q, \Gamma \vdash R}{P \longrightarrow Q, \Gamma \vdash R}\ \texttt{impE} \qquad \frac{P \longrightarrow Q, Q \longrightarrow P, \Gamma \vdash R}{P \longleftrightarrow Q, \Gamma \vdash R}\ \texttt{iffE}$$

$$\frac{P \wedge Q \qquad [\![P;\ Q]\!] \Longrightarrow R}{R}\ \texttt{conjE} \qquad \frac{P \vee Q \qquad P \Longrightarrow R \qquad Q \Longrightarrow R}{R}\ \texttt{disjE}$$

$$\frac{P \longrightarrow Q \qquad P \qquad Q \Longrightarrow R}{R}\ \texttt{impE} \qquad \frac{P \longleftrightarrow Q \qquad [\![P \longrightarrow Q;\ Q \longrightarrow P]\!] \Longrightarrow R}{R}\ \texttt{iffE}$$

- Elimination rules use erule tactical **apply** `(erule conjE)`.
- Find first hypothesis matching first premise of elimination rule.
- Replace subgoal with remaining premises: $[\![P;\ Q]\!] \Longrightarrow R$.

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

asm asm

impE

conjE

impI

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

asm asm

impE

conjE

impI

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

asm

$P \Longrightarrow P$

asm

$[\![P; Q]\!] \Longrightarrow Q$

impE

$[\![(P \longrightarrow Q); P]\!] \Longrightarrow Q$

conjE

$P \wedge (P \longrightarrow Q) \Longrightarrow Q$

impI

$P \wedge (P \longrightarrow Q) \longrightarrow Q$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{\dfrac{\overline{P \Longrightarrow P}\ \text{asm} \qquad \overline{[\![P;\ Q]\!] \Longrightarrow Q}\ \text{asm}}{[\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q}\ \text{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\ \text{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\ \text{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{\dfrac{\overline{P \Longrightarrow P}\ \texttt{asm} \qquad \overline{[\![P;\ Q]\!] \Longrightarrow Q}\ \texttt{asm}}{[\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q}\ \texttt{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\ \texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\ \texttt{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\frac{P \wedge (P \longrightarrow Q) \Longrightarrow Q}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\ \texttt{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\;\texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\;\texttt{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule impE)
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{[\![ (P \longrightarrow Q);\; P ]\!] \Longrightarrow Q}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\;\texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\;\texttt{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{P \Longrightarrow P}\,\texttt{asm} \qquad \dfrac{}{[\![P;\ Q]\!] \Longrightarrow Q}\,\texttt{asm}}{[\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q}\,\texttt{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\,\texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\,\texttt{impI}
$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P \Longrightarrow P}\,\text{asm} \qquad \dfrac{}{[\![P;\ Q]\!] \Longrightarrow Q}\,\text{asm}}{[\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q}\,\texttt{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\,\texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\,\texttt{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
   (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{P \Longrightarrow P}\,\texttt{asm} \qquad \dfrac{}{[\![P;\; Q]\!] \Longrightarrow Q}\,\texttt{asm}}{[\![(P \longrightarrow Q);\; P]\!] \Longrightarrow Q}\,\texttt{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\,\texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\,\texttt{impI}
$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{\dfrac{\dfrac{-}{P \Longrightarrow P}\,\texttt{asm} \qquad \dfrac{-}{[\![P;\ Q]\!] \Longrightarrow Q}\,\texttt{asm}}{[\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q}\,\texttt{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\,\texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\,\texttt{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$
\dfrac{
  \dfrac{
    \dfrac{}{P \Longrightarrow P}\,\text{asm}
    \qquad
    \dfrac{}{[\![P;\ Q]\!] \Longrightarrow Q}\,\text{asm}
  }{
    [\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q
  }\,\text{impE}
}{
  \dfrac{P \wedge (P \longrightarrow Q) \Longrightarrow Q}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\,\text{impI}
}\,\text{conjE}
$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{\dfrac{\dfrac{-}{P \Longrightarrow P}\,\texttt{asm} \qquad \dfrac{-}{[\![P;\ Q]\!] \Longrightarrow Q}\,\texttt{asm}}{[\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q}\,\texttt{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\,\texttt{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\,\texttt{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (1)

How do we prove $P \wedge (P \longrightarrow Q) \longrightarrow Q$?

## Natural Deduction Proof

$$\dfrac{\dfrac{\dfrac{\dfrac{-}{P \Longrightarrow P}\ \text{asm} \qquad \dfrac{-}{[\![P;\ Q]\!] \Longrightarrow Q}\ \text{asm}}{[\![(P \longrightarrow Q);\ P]\!] \Longrightarrow Q}\ \text{impE}}{P \wedge (P \longrightarrow Q) \Longrightarrow Q}\ \text{conjE}}{P \wedge (P \longrightarrow Q) \longrightarrow Q}\ \text{impI}$$

## Proof Script

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
  apply (rule impI)
  apply (erule conjE)
   apply (erule imp)E
    (* Subgoal 1 *)
    apply assumption
   (* Subgoal 2 *)
   apply assumption
   done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

Proof Script

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$(P \wedge P) \longleftrightarrow P$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\frac{}{(P \wedge P) \longleftrightarrow P}\ \mathtt{iffI}$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{P \wedge P \Longrightarrow P \qquad\qquad P \Longrightarrow P \wedge P}{(P \wedge P) \longleftrightarrow P}\ \texttt{iffI}$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{}{\llbracket P;\ P \rrbracket \Longrightarrow P}\,\texttt{asm}}{P \wedge P \Longrightarrow P}\,\texttt{conjE} \qquad \dfrac{\dfrac{}{P \Longrightarrow P}\,\texttt{asm} \quad \dfrac{}{P \Longrightarrow P}\,\texttt{asm}}{P \Longrightarrow P \wedge P}\,\texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\,\texttt{iffI}$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{}{[\![P;\ P]\!] \Longrightarrow P}\,\mathtt{asm}}{P \wedge P \Longrightarrow P}\,\mathtt{conjE} \qquad \dfrac{\dfrac{}{P \Longrightarrow P}\,\mathtt{asm} \quad \dfrac{}{P \Longrightarrow P}\,\mathtt{asm}}{P \Longrightarrow P \wedge P}\,\mathtt{conjI}}{(P \wedge P) \longleftrightarrow P}\,\mathtt{iffI}$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{}{\llbracket P;\ P \rrbracket \Longrightarrow P}\,\texttt{asm}}{P \wedge P \Longrightarrow P}\,\texttt{conjE} \qquad \dfrac{\dfrac{}{P \Longrightarrow P}\,\texttt{asm} \quad \dfrac{}{P \Longrightarrow P}\,\texttt{asm}}{P \Longrightarrow P \wedge P}\,\texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\,\texttt{iffI}$$

## Proof Script

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{}{[\![P;\ P]\!] \Longrightarrow P}\ \texttt{asm}}{P \wedge P \Longrightarrow P}\ \texttt{conjE} \qquad \dfrac{\dfrac{}{P \Longrightarrow P}\ \texttt{asm} \quad \dfrac{}{P \Longrightarrow P}\ \texttt{asm}}{P \Longrightarrow P \wedge P}\ \texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\ \texttt{iffI}$$

## Proof Script

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$
\dfrac{
  \dfrac{\dfrac{}{[\![P;\; P]\!] \Longrightarrow P}\,\texttt{asm}}{P \wedge P \Longrightarrow P}\,\texttt{conjE}
  \qquad
  \dfrac{\dfrac{}{P \Longrightarrow P}\,\texttt{asm} \quad \dfrac{}{P \Longrightarrow P}\,\texttt{asm}}{P \Longrightarrow P \wedge P}\,\texttt{conjI}
}{(P \wedge P) \longleftrightarrow P}\,\texttt{iffI}
$$

### Proof Script

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{}{[\![P;\ P]\!] \Longrightarrow P}\ \texttt{asm}}{P \wedge P \Longrightarrow P}\ \texttt{conjE} \qquad \dfrac{\dfrac{}{P \Longrightarrow P}\ \texttt{asm} \quad \dfrac{}{P \Longrightarrow P}\ \texttt{asm}}{P \Longrightarrow P \wedge P}\ \texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\ \texttt{iffI}$$

## Proof Script

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$
\dfrac{
  \dfrac{
    \dfrac{}{\llbracket P;\ P \rrbracket \Longrightarrow P}\ \texttt{asm}
  }{P \wedge P \Longrightarrow P}\ \texttt{conjE}
  \qquad
  \dfrac{
    \dfrac{}{P \Longrightarrow P}\ \texttt{asm}
    \quad
    \dfrac{}{P \Longrightarrow P}\ \texttt{asm}
  }{P \Longrightarrow P \wedge P}\ \texttt{conjI}
}{(P \wedge P) \longleftrightarrow P}\ \texttt{iffI}
$$

## Proof Script

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{}{[\![P;\ P]\!] \Longrightarrow P}\ \texttt{asm}}{P \wedge P \Longrightarrow P}\ \texttt{conjE} \qquad \dfrac{\dfrac{}{P \Longrightarrow P}\ \texttt{asm} \quad \dfrac{}{P \Longrightarrow P}\ \texttt{asm}}{P \Longrightarrow P \wedge P}\ \texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\ \texttt{iffI}$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{}{[\![P;\ P]\!] \Longrightarrow P}\,\texttt{asm}}{P \wedge P \Longrightarrow P}\,\texttt{conjE} \qquad \dfrac{\dfrac{}{P \Longrightarrow P}\,\texttt{asm} \qquad \dfrac{}{P \Longrightarrow P}\,\texttt{asm}}{P \Longrightarrow P \wedge P}\,\texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\,\texttt{iffI}$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{-}{[\![P;\ P]\!] \Longrightarrow P}\,\texttt{asm}}{P \wedge P \Longrightarrow P}\,\texttt{conjE} \qquad \dfrac{\dfrac{-}{P \Longrightarrow P}\,\texttt{asm} \quad \dfrac{-}{P \Longrightarrow P}\,\texttt{asm}}{P \Longrightarrow P \wedge P}\,\texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\,\texttt{iffI}$$

**Proof Script**

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Example: Proof by Elimination (2): $(P \wedge P) \longleftrightarrow P$

$$\dfrac{\dfrac{\dfrac{-}{\llbracket P;\ P\rrbracket \Longrightarrow P}\,\texttt{asm}}{P \wedge P \Longrightarrow P}\,\texttt{conjE} \qquad \dfrac{\dfrac{-}{P \Longrightarrow P}\,\texttt{asm} \quad \dfrac{-}{P \Longrightarrow P}\,\texttt{asm}}{P \Longrightarrow P \wedge P}\,\texttt{conjI}}{(P \wedge P) \longleftrightarrow P}\,\texttt{iffI}$$

## Proof Script

```
lemma "(P ∧ P) ⟷ P"
  apply (rule iffI)
   apply (erule conjE)
   apply assumption
  apply (rule conjI)
    apply assumption
   apply assumption
  done
```

# Outline

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state
- A little too verbose for simple predicates

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state
- A little too verbose for simple predicates

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Natural Deduction in Isar

```
lemma "P ∧ (P ⟶ Q) ⟶ Q"
proof (rule impI)
  assume "P ∧ (P ⟶ Q)"
  hence "P ⟶ Q" "P"
    by (erule_tac conjE, simp_all)
  thus "Q"
    by (erule_tac impE, simp_all)
qed
```

- Benefit of being readable without seeing the proof state.
- A little too verbose for simple predicates.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

lemma "P $\longrightarrow$ (P $\wedge$ (P $\vee$ Q))" by blast
lemma "P $\wedge$ (P $\longrightarrow$ Q) $\longrightarrow$ Q" by blast

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Automating Natural Deduction

- Isn't all this low level deduction too much work?
- Fortunately, Isabelle automates natural deduction using the blast tactic.
- Tableaux prover: represent proof tree structure.

**Example**

```
lemma "P ⟶ (P ∧ (P ∨ Q))" by blast
lemma "P ∧ (P ⟶ Q) ⟶ Q" by blast
```

- Blast searches proof tree using introduction and elimination rules.
- `blast intro: thms elim: thms`, or `[intro]`, `[elim]`, and `[dest]`.
- Safe rules marked with "!", e.g. `[intro!]`.
- Unsafe rules applied only when no alternative.
- Blast is one-shot: all or nothing.

# Combining Deduction and Simplification with Auto

# Combining Deduction and Simplification with Auto

- `auto` is a powerful tactic combining deduction (e.g., `blast`) and `simp`.
- Not one-shot: applies safe rules repeatedly, simplifying if possible.
- Won't lose information as long as the specified safe rules are truly safe.
- General technique for breaking a complex goal into several parts.

```
lemma "(x::nat) = y² ⟶ x ≥ y ∧ x ≥ 0"
  by (auto simp add: power2_nat_le_imp_le)
```

- There are also one-shot variants: `force` and `fastforce`.
- `auto` can produce unpredictable and bizarre results.
- Still need manual deduction to understand why automated proof fails.

# Combining Deduction and Simplification with Auto

- `auto` is a powerful tactic combining deduction (e.g., `blast`) and `simp`.
- Not one-shot: applies safe rules repeatedly, simplifying if possible.
- Won't lose information as long as the specified safe rules are truly safe.
- General technique for breaking a complex goal into several parts.

```
lemma "(x::nat) = y² ⟶ x ≥ y ∧ x ≥ 0"
  by (auto simp add: power2_nat_le_imp_le)
```

- There are also one-shot variants: `force` and `fastforce`.
- `auto` can produce unpredictable and bizarre results.
- Still need manual deduction to understand why automated proof fails.

# Combining Deduction and Simplification with Auto

- `auto` is a powerful tactic combining deduction (e.g., `blast`) and `simp`.
- Not one-shot: applies safe rules repeatedly, simplifying if possible.
- Won't lose information as long as the specified safe rules are truly safe.
- General technique for breaking a complex goal into several parts.

```
lemma "(x::nat) = y² ⟶ x ≥ y ∧ x ≥ 0"
  by (auto simp add: power2_nat_le_imp_le)
```

- There are also one-shot variants: `force` and `fastforce`.
- `auto` can produce unpredictable and bizarre results.
- Still need manual deduction to understand why automated proof fails.

# Combining Deduction and Simplification with Auto

- `auto` is a powerful tactic combining deduction (e.g., `blast`) and `simp`.
- Not one-shot: applies safe rules repeatedly, simplifying if possible.
- Won't lose information as long as the specified safe rules are truly safe.
- General technique for breaking a complex goal into several parts.

```
lemma "(x::nat) = y² ⟶ x ≥ y ∧ x ≥ 0"
  by (auto simp add: power2_nat_le_imp_le)
```

- There are also one-shot variants: `force` and `fastforce`.
- `auto` can produce unpredictable and bizarre results.
- Still need manual deduction to understand why automated proof fails.

# Combining Deduction and Simplification with Auto

- `auto` is a powerful tactic combining deduction (e.g., `blast`) and `simp`.
- Not one-shot: applies safe rules repeatedly, simplifying if possible.
- Won't lose information as long as the specified safe rules are truly safe.
- General technique for breaking a complex goal into several parts.

```
lemma "(x::nat) = y² ⟶ x ≥ y ∧ x ≥ 0"
  by (auto simp add: power2_nat_le_imp_le)
```

- There are also one-shot variants: `force` and `fastforce`.
- `auto` can produce unpredictable and bizarre results.
- Still need manual deduction to understand why automated proof fails.

# Combining Deduction and Simplification with Auto

- `auto` is a powerful tactic combining deduction (e.g., `blast`) and `simp`.
- Not one-shot: applies safe rules repeatedly, simplifying if possible.
- Won't lose information as long as the specified safe rules are truly safe.
- General technique for breaking a complex goal into several parts.

```
lemma "(x::nat) = y² ⟶ x ≥ y ∧ x ≥ 0"
  by (auto simp add: power2_nat_le_imp_le)
```

- There are also one-shot variants: `force` and `fastforce`.
- `auto` can produce unpredictable and bizarre results.
- Still need manual deduction to understand why automated proof fails.

# Combining Deduction and Simplification with Auto

- `auto` is a powerful tactic combining deduction (e.g., `blast`) and `simp`.
- Not one-shot: applies safe rules repeatedly, simplifying if possible.
- Won't lose information as long as the specified safe rules are truly safe.
- General technique for breaking a complex goal into several parts.

```
lemma "(x::nat) = y² ⟶ x ≥ y ∧ x ≥ 0"
  by (auto simp add: power2_nat_le_imp_le)
```

- There are also one-shot variants: `force` and `fastforce`.
- `auto` can produce unpredictable and bizarre results.
- Still need manual deduction to understand why automated proof fails.

# Applications

- We looked at automating natural deduction in the propositional calculus.
- We demonstrated a far more general technique.
- We define bespoke logics in Isabelle and then reason about them.
- In particular, we can program verification (e.g., Hoare logic).
- Natural deduction is an important weapon in the proof arsenal.

# Applications

- We looked at automating natural deduction in the propositional calculus.
- We demonstrated a far more general technique.
- We define bespoke logics in Isabelle and then reason about them.
- In particular, we can program verification (e.g., Hoare logic).
- Natural deduction is an important weapon in the proof arsenal.

# Applications

- We looked at automating natural deduction in the propositional calculus.
- We demonstrated a far more general technique.
- We define bespoke logics in Isabelle and then reason about them.
- In particular, we can program verification (e.g., Hoare logic).
- Natural deduction is an important weapon in the proof arsenal.

# Applications

- We looked at automating natural deduction in the propositional calculus.
- We demonstrated a far more general technique.
- We define bespoke logics in Isabelle and then reason about them.
- In particular, we can program verification (e.g., Hoare logic).
- Natural deduction is an important weapon in the proof arsenal.

# Applications

- We looked at automating natural deduction in the propositional calculus.
- We demonstrated a far more general technique.
- We define bespoke logics in Isabelle and then reason about them.
- In particular, we can program verification (e.g., Hoare logic).
- Natural deduction is an important weapon in the proof arsenal.

# Conclusion

## This Lecture

- Apply-style proof scripts.
- Natural deduction in Isabelle/HOL.
- The classical reasoner.

## Next Lecture

- Predicate calculus (quantifiers etc.).

# Conclusion

## This Lecture

- Apply-style proof scripts.
- Natural deduction in Isabelle/HOL.
- The classical reasoner.

## Next Lecture

- Predicate calculus (quantifiers etc.).

# Conclusion

## This Lecture

- Apply-style proof scripts.
- Natural deduction in Isabelle/HOL.
- The classical reasoner.

## Next Lecture

- Predicate calculus (quantifiers etc.).

# Conclusion

## This Lecture

- Apply-style proof scripts.
- Natural deduction in Isabelle/HOL.
- The classical reasoner.

## Next Lecture

- Predicate calculus (quantifiers etc.).

# Conclusion

## This Lecture

- Apply-style proof scripts.
- Natural deduction in Isabelle/HOL.
- The classical reasoner.

## Next Lecture

- Predicate calculus (quantifiers etc.).

# Conclusion

## This Lecture

- Apply-style proof scripts.
- Natural deduction in Isabelle/HOL.
- The classical reasoner.

## Next Lecture

- Predicate calculus (quantifiers etc.).

# Conclusion

## This Lecture

- Apply-style proof scripts.
- Natural deduction in Isabelle/HOL.
- The classical reasoner.

## Next Lecture

- Predicate calculus (quantifiers etc.).