Disjvnction

What do we mean by disjunction

- $P \lor Q$ is true if P is true, or if Q is true, or if both are true.
  - In logic, disjunctions typically use the word “or” to mean that either one or both of the propositions can be true
  - Exclusive-or (xor) is used to mean exactly one of the two propositions is true
  - In some contexts, a disjunction might take more than two propositions, in which case the disjunction is true if *at least* one of the propositions is true
- Consider the statement “it is raining or the sprinkler is on”, this proposition is true if:
  - It is raining, and the sprinkler is off
  - It is not raining, but the sprinkler is on
  - It is raining, and the sprinkler is on

Story Time (Disjunction vs Conjunction)

- Make sure you use $\vee$ and $\wedge$ correctly
- A researcher was working on a function with an input *x* that had to have a magnitude greater than one
  - She wrote her pre-condition as
    - $x < -1 \wedge 1 < x$
  - She was able to prove her post-condition, but that proof was meaningless
    - Precondition should've been $x < -1 \vee 1 < x$
- If you had an input p that has to be between 0 and 1, which of these is correct?
  - $0 < p \wedge p < 1$
  - $0 < p \vee p < 1$

Introduction rules

- Unlike conjunction which requires both propositions to be true, disjunction only requires one proposition to be true, so we have two introduction rules:

```
{ P } Q : Prop, pfP: P
---------------------- (or.intro_left)
         P ∨ Q


P { Q } : Prop, pfQ: Q
----------------------- (or.intro_right)
         P ∨ Q
```

Introduction rules in Lean

- Lean provides the or.intro_left and or.intro_right rules

```
theorem goodSide : 0 = 0 ∨ 0 = 1 :=
  or.intro_left (0=1) (eq.refl 0)


example : P ∨ false :=
  or.intro_left false pfP

example : false ∨ P :=
  or.intro_right false pfP
```

An aside on or_intro.left

- Right click on or_intro.left and choose “peek definition”:

```
def or.intro_left {a : Prop} (b : Prop) (ha : a) : or a b :=
or.inl ha
```

- Note that
  - or_intro.left has 3 arguments, the first of which is implicit, and the second two of which are explicit.
  - The first two arguments are propositions and the third is a proof of the first proposition (which is how the first argument is inferred)
  - or_intro.left has a return type of “`or a b`”

Introduction rules in Lean (revisited)

- Lean provides the or.inl and or.inr rules

```
theorem goodSide : 0 = 0 ∨ 0 = 1 :=
  or.inl (eq.refl 0)


example : P ∨ false :=
  or.inl pfP

example : false ∨ P :=
  or.inr pfP
```

Exercise

- Prove 0 = 1 ∨ 0 = 0 two different ways, once with or.intro_right and once with or.inr

```
example: 0 = 1 ∨ 0 = 0 :=
  or.intro_right (0 = 1) (eq.refl 0)

example: 0 = 1 ∨ 0 = 0 :=
  or.inr (eq.refl 0)
```

Disjunction absorption
(or the "zero" of disjunction)

- In multiplication, 0 is the absorption element, because anything times 0 is 0
- What is the absorption element in disjunction? I.e., what is the element X such that $P \vee X = X$ for all propositions P?

Disjunction with true

- See disjunction_properties.lean for proof of right absorption
- Exercise: Prove that true or Q is always true as well.

Elimination rule

- The elimination rule for disjunction has some nuance:
  - If P ∨ Q, then if both P → R and Q → R, then R

```
 pfPQ: P ∨ Q, pfPR: P → R, pfQR: Q → R
---------------------------------------- ∨-elim
                   R
```

- Example:
  - When it's raining, the grass is wet (P → R)
  - When the sprinkler's on, the grass is wet (Q → R)
  - It's raining, or the sprinkler's on (P ∨ Q)
  - Therefore, the grass must be wet (R)
- See Lean for examples

Cases

- We’ve already seen cases, but now let’s examine it more deeply with respect to disjunction:

```
theorem wet''' :
  ∀ R S W: Prop,
    (R ∨ S) → (R → W) → (S → W) → W :=
begin
  assume R S W pfRorS r2w s2w,
  cases pfRorS with pfR pfS,
    exact r2w pfR, -- case when it’s raining
    exact s2w pfS, -- case when the sprinkler’s on
end
```

Disjunction Identity

- In multiplication, 1 is the identity element, because x times 1 is x for any value of x
- In addition, 0 is the identity element, because x plus 0 is x for any value of x
- What is the identity element for disjunction?

Right Identity

- See disjunction_properties.lean for proof of right identity

Exercise

- Prove that false is also a left identity

Disjunction Syllogism

```
theorem disjunctiveSyllogism :
  ∀(P Q: Prop), P ∨ Q → ¬Q → P :=
begin
  intros P Q pfPOrQ pfNotQ, -- assumptions
  cases pfPOrQ with pfP pfQ, -- now by cases
    assumption, -- case where p is true,
    exact false.elim (pfNotQ pfQ) -- or q true
end
```

De Morgan's Laws

- First law: $\neg(P \lor Q) \leftrightarrow \neg P \land \neg Q$
  - If it's not true that P or Q is true, then it's true that P is not true and Q is not true
  - If it's not true that it's raining or the sprinkler is on, then it's true that it's not raining and the sprinkler is not on
  - Homework #4
- Second law: $\neg(P \land Q) \leftrightarrow \neg P \lor \neg Q$
  - If it's not true that P and Q is true, then it's true that P is not true or Q is not true
  - If it's not true that it's raining and the sprinkler is on, then it's true that either it's not raining, or the sprinkler is off, or both
  - See Lean for proof

Disjunction Properties

- Reflexive? Is $A \vee A$ true?
  - No
- Symmetric/Commutative? Does $(A \vee B) \rightarrow (B \vee A)$?
  - Yes
- Associative? Does $(A \vee B) \vee C \leftrightarrow A \vee (B \vee C)$?
  - Yes
- Transitive? Does $(A \vee B) \rightarrow (B \vee C) \rightarrow (A \vee C)$?
  - No

Distribution rules for conjunction/disjunction

- Distribution over conjunction
  - $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$

- Distribution over disjunction
  - $A \wedge (B \vee C) \leftrightarrow (A \wedge B) \vee (A \wedge C)$

- See disjunction_properties.lean

Fin