**Module 2**

* A logic consists of:
  + Syntax
  + Semantics
  + Proof theory(s)
* **Proof theory** – methods that perform mechanical manipulations on strings of symbols
  + Is not concerned with the meanings of sentences; only treats them as strings of characters
  + Based on pattern matching
* |= → used in semantics; “entails”
* |− → used in proof theory; “proves”

|  |  |  |
| --- | --- | --- |
| **Syntax** | **Semantics** | **Proof Theory** |
| ⇒ | |= | |− |
| ⇔ | <≡> | ↔ |

* **Soundness** – a proof theory is sound if P1 … Pn |− Q (proof) then P1 … Pn |= Q (valid)
  + If it can be proved then it must be true
* **Completeness** – a proof theory is complete if P1 … Pn |= Q (valid) then P1 … Pn |− Q (proof)
  + If it is true then it can be proved
* **Propositional logic – syntax**
  + Symbols
    - 2 constants: true & false
    - Proposition symbols (lower case letters)
    - Connectives:
      * ¬, ∧, ∨, ⇒, ⇔ (← in the order of precedence)
    - Brackets
  + All binary logical connectives are right associative
    - E.g. a ⇒ b ⇒ c is equivalent to a ⇒ (b ⇒ c)
  + **Well-formed formula**:
    - Proposition symbols + constants true & false are formulas (prime propositions)
    - If P & Q are formulas, these are formulas (compound propositions):
      * (¬P), (P ∧ Q), (P ∨ Q), (P ⇒ Q), (P ⇔ Q)
      * No other expressions are formulas
  + In P ∧ Q, P & Q are conjuncts
  + In P ∨ Q, P & Q are disjuncts
  + In P ⇒ Q, P is the premise/hypothesis, Q is the consequent/conclusion
  + The contrapositive of P ⇒ Q is ¬Q ⇒ ¬P
* Formalizing natural language
  + Proposition symbols represent declarative sentences
  + Beware of ambiguity in language
* **Propositional logic – semantics**
  + Semantics means “meaning”
  + Syntax for propositional logic is the domain of the semantic function
  + Truth values in classical logic consist of T and F
    - The set of truth values is the range of the semantic function
    - Tr = {T, F}
  + The semantic function maps expressions of syntax to truth values
  + **Boolean valuation** – function from the set of formulas in propositional logic → Tr
    - Underline is a semantic function that means “meaning of the formula”
    - NOT, AND, OR, IMP, IFF are functions that are the meanings of their corresponding symbols
    - I.e. false = F, true = T, ¬P = NOT(P), etc.
    - Only need to describe the association of truth values w/ proposition symbols
  + **Truth table**
    - Each row is a possible Boolean valuation
    - Each cell contains the truth value for the subformula w/ row’s Boolean valuation
  + A formula P is **satisfiable** if there exists a Boolean valuation such that P = T
  + A formula P is a **tautology (valid)** if P = T for all Boolean valuations
    - P is a tautology → **|= P**
  + A propositional formula A is a **contradiction** if A = F for all Boolean valuations
  + A **contingent** formula is neither a tautology nor contradiction
  + A formula P **logically implies** a formula Q iff for all Boolean valuations, if P = T then Q = T
    - i.e. **P |= Q** or **|= P ⇒ Q**
    - i.e. the implication P ⇒ Q is a tautology
    - i.e. P ⇒ Q is a **valid argument**
  + Generalize: a set of formulas P1 … Pn **logically implies** a formula Q iff for all Boolean valuations, if P1 = T and P2 = T and … then Q = T
    - i.e. **P1 … Pn |= Q**
    - i.e. **P1 ∧ … ∧ Pn |= Q**
    - In a truth table, find all rows where all premises have truth values of T
      * The conclusion should also have truth values of T in all of these rows
  + Two formulas are **logically equivalent** iff for all possible Boolean valuations, P = Q
    - i.e. **P <≡> Q**
    - i.e. **|= P ⇔ Q** (material equivalence)
    - i.e. For all Boolean valuations, P = T iff Q = T
    - To show that P <≡/≡> Q, find a B. v. where P ≠ Q
  + A collection of formulas is **consistent** if there exists a B. v. in which they can all be true simultaneously
    - i.e. the conjunction of the formulas (P1 ∧ P2 ∧ … ) is satisfiable
    - Thus if the set of premises is not consistent, they can logically imply anything (**false implies anything**)