**Ch 1 - Atomic Sentences**

* FOL is a family of languages
  + similar grammar
  + sharing of important vocabulary items (connectives, quantifiers)
* each language in the family can differ in the atomic sentences used (specific vocabulary used to form most basic sentences)

**Atomic Sentences**

* in English: **names** connected by a **predicate**
* in FOL
  + combination of **names** (or individual **constants**) and **predicates**
* different versions of FOL have available different names and predicates
* **examples**
  + Cube(b)
  + Larger(c,f)
  + Between(b,c,d)

**Individual Constants**

* symbols used to refer to some fixed individual object
* FOL analogue of names, but usually not capitalized
* **examples**
  + max
  + 1
* **requirements**
  + an individual constant refers to exactly one object
  + an object can be referred to by more than one name
  + a name must refer to an actually existing object

**Predicate Symbols**

* symbols used to express some property of objects or relation between objects
* aka **relation symbols**
* when combined with names form atomic sentences
* don’t correspond exactly with predicates of English grammar
* **examples**
  + Max likes Claire.
    - in English: subject-predicate sentence
    - in FOL, this claim involves two **logical subjects,** *Max* and *Claire* and a **predicate** *likes* that expresses a relation between the **referents** of the names
* in FOL
  + atomic sentences often have two or more logical subjects and the predicate is, so to speak, whatever is left
  + **logical subjects** are called **arguments of the predicate**
  + each predicate has a **fixed number of arguments,** aka **fixed arity:** how many individual constants the predicate symbol needs in order to form a sentence
  + unary, binary, ternary, etc: origin or term “arity”
* **examples**
  + arity 1
    - Home(max)
  + arity 2
    - Taller(claire, max)
* FOL assumes that every predicate is interpreted by a **determinate property** or relation
  + determinate property: property for which, given an object, there is a definite fact of the matter whether or not the object has the property

**Atomic Sentences**

* simplest kinds of claims: single predicate and appropriate number of individual constants; this is called an **atomic sentence**
  + Taller(claire, max), Cube(a) are atomic sentences provided the names and predicate symbols in question are part of the vocabulary of our language
* **example**
  + **a=b**
    - infix notation
    - predicate symbol = appears between its two arguments
* order of names in atomic sentence is important
  + Taller(claire,max) different from Taller(max, claire)
* **predicates: designate properties**
* **names: designate objects**
* **sentences: make claims (express propositions)**
* **claim: either true or false**
  + which of these it is is called its **truth value**
* therefore, **each atomic sentence of FOL must express a claim that is either true or false**

**General First-Order Languages**

* in general, when designing a first-order language, we try to economize on the predicates by introducing more flexible ones, producing a more expressive language that makes the logical relations between various claims more **perspicuous**
* **object:** anything we can make claims about

**Function Symbols**

* another type of expressions, other than names and predicates, that can appear in atomic sentences is called a **function symbol**
  + allow us to form name-like terms from names and other name-like terms
* **complex terms:** typically formed by putting a function symbol of arity n in front of n terms (simple or complex)
* **example**
  + *Max’s father*
  + ***father(max)***
    - father is a function symbol
    - result is a **complex term** that we use to refer to the father of the person referred to by the name *max*
    - is a **unary symbol** because takes one argument
* **examples**
  + *father(father(max))*
* *father(max)* can be used just like a name: *Taller(father(max),max)*
* **difference between function symbols and predicates**
  + when we combine a unary function symbol with a term you do not get a sentence, but another term: something that refers to an object of some sort
  + function symbols can be reapplied over and over again
  + **we always capitalize predicates of FOL but leave function symbols and names in lower case**
* **examples binary function symbols**
  + ***sum(4,5)***
* it is required in FOL that every complex term refer to exactly one object

**FOL of Set Theory**

* FOL was initially developed for use in mathematics
* most FOL are those associated with various branches of mathematics
* **example:** language of set theory
* has only two predicates, both binary
  + identity symbol, =
  + set membership, ∈
* standard use of these predicates is infix
* there are no function symbols in the language of set theory

**FOL of Arithmetic**

* allows us to express statements about the natural numbers
* has function symbols
* several more or less equivalent ways of setting up this language
* Our setup
  + two names: 0 and 1
  + two binary relation symbols: = and <
  + two binary function symbols: + and ×
* atomic sentences can then be built up out of these symbols
* infix notation used for all the symbols
* there are infinitely many terms in this language
  + 0, 1, 1+1, (1+1)+1, …
  + and so an infinite number of atomic sentences
* every natural number is named by some term in the language
* **how can we specify the set of terms in a precise way?**
  + using **inductive definition**
  + **Definition:** the terms of first-order arithmetic are formed in the following way
    - 1) the names 0, 1 are terms
    - 2) if t1 and t2 are terms, then expressions t1+t2 and t1×t2 are also terms
    - nothing is a term unless it can be obtained by repeated application of 1) and 2)
* **example atomic sentence**
  + (1×1)<(1+1)