Chapter 3 – Using Data Structures

* Structures and Trees
  + Tree- each function is a node, and the components are branches
* Lists
  + List- very common data structures in non-numeric programming
  + List is an ordered sequence of elements that can have any length
* Recursive Search
  + The ability to search inside a Prolog structure, which may have other structures as its components, to find some desired piece of information
* Mapping
  + Given a Prolog structure, we frequently wish to construct a new structure that is like the old one but changed in some way. We traverse the old structure component-by-component, and construct the components of the new structure
* Recursive Comparison
  + Prolog provides predicates to compare integers
  + Comparing structures is generally more complicated, because it is necessary to consider all the individual components
  + When the components may themselves be structures, the comparison may have to be recursive
* Joining Structures Together
  + List processing predicate append is used to join two lists together to form another, new, list
  + Append is most often used to create a new list from concatenating two others
* Accumulators
  + An argument of the predicate to represent “the answer so far”
  + Used to traverse a Prolog structure and calculate a result that depends on what was found in the structure
* Difference Structures
  + Allow us to generate elements in the same order as the original list, when the elements are in reverse order