Notes lecture 3

* Predicates types:
  + Ideal implementation
    - **Deterministic** = has only one solution
    - **Non- deterministic** = has several solutions
  + In SWI-Prolog:
    - **Deterministic** = always succeeds exactly once and does not leave a choicepoint.
    - **Semi-deterministic** = succeeds at most once

**+** If it succeeds it does not leave a choicepoint.

* + - **Non-deterministic** = no claims are made on the number of solutions and whether the predicate leaves a choicepoint on the last solution.
* Collecting all solutions:

|  |  |  |  |
| --- | --- | --- | --- |
|  | *findall()* | *setof()* | *bagof()* |
| Removes duplicates | No | Yes | No |
| Sorted results | No | Yes | No |
| any variable in goal not in the first argument | Not instantiated | Separate solutions | Separate solutions |

* Negation:
  + not(sth)
  + \+ sth
* Lists
  + Heterogeneous = elements in them are of different types
  + Head & Tail:
    - The following lists are equivalent:
      * [a b c]
      * [a | [b, c]]
      * [a | [b | [c]]]
      * [a | [b | [c | []]]]
  + Because a list is a *recursive* data structure, *recursive algorithms* are needed to process it, which generally need two clauses
    - what to do with an empty list
    - what to do with a non-empty list (which can be broken down in its head and its tail) then repeat process for the tail