Funções Prolog

length\2

True if *Length* represents the number of elements in *List*.

```
length([1,1,3], A)
```

This predicate is a true relation and can be used to produce a list (holding variables) of length *Length*.

```
length(L, 3)
```

```
A is 0, length(L, 3), maplist(=(A), L)
```

The predicate is non-deterministic, producing lists of increasing length if *List* is a *partial list* and *Length* is a variable.

```
length(L, R)
```

append\3

append(*?List1*, *?List2*, *?List1AndList2*)

List1AndList2 is the concatenation of *List1* and *List2*

```
append([a,b], [c], X).
```

```
append(X, [Last], [a,b,c]).
```

```
append([a,b], More, List).
```

findall\3

Create a list of the instantiations *Template* gets successively on backtracking over *Goal* and unify the result with *Bag*. Succeeds with an empty list if *Goal* has no solutions.

```
foo(a, b, c).

foo(a, b, d).

foo(b, c, e).

foo(b, c, f).

foo(c, c, g).

findall(C, foo(A,B,C), L)
```