Some sicstus built-ins

`append(?Prefix, ?Suffix, ?Combined)`

*Combined* is the combined list of the elements in *Prefix* followed by the elements in *Suffix*.

`findall(?Template,:Goal,?Bag)`

*Bag* is a list of instances of *Template* in all proofs of *Goal* found by Prolog. The list may be empty and all variables are taken as being existentially quantified.

`setof(?Template,:Goal,?Set)`

Read this as “*Set* is the set of all instances of *Template* such that *Goal* is satisfied, where that set is non-empty”. The term *Goal* specifies a goal or goals as in `call(Goal)`

`bagof(?Template,:Goal,?Bag)`

This is exactly the same as `setof/3` except that the list (or alternative lists) returned will not be ordered, and may contain duplicates.

`Z is X`  
*X*, which must be an arithmetic expression, is evaluated and the result is unified with *Z*.

`X =:= Y`  
The numeric values of *X* and *Y* are equal.

`+Term =.. ?List  ?Term =.. +List`  
*List* is a list whose head is the atom corresponding to the principal functor of *Term*, and whose tail is a list of the arguments of *Term*.

`Term1 == Term2`  
The terms currently instantiating *Term1* and *Term2* are literally identical.

`var(?X)`

Tests whether *X* is currently uninstantiated.

`atom(?X)`

Tests whether *X* is currently instantiated to an atom.

`functor(+Term,?Name,?Arity)  functor(?Term,+Name,+Arity)`

The principal functor of term *Term* has name *Name* and arity *Arity*

`arg(+ArgNo,+Term,?Arg)`

*Arg* is the argument *ArgNo* of the compound term *Term*. The arguments are numbered from 1 upwards.