Turbo Prolog 2.0. Lists.

General.

[1,2,3,6,9,3,4]
[3.2,4.6,1.1,2.64,100.2] ["YESTERDAY","TODAY","TOMORROW"]

_____

List	***Head***	***Tail***
[1,2,3,4,5]	1	[2,3,4,5]
['s', 'k, 'y']	's'	['k', 'y']
[cat]	cat	[]
	It is not certain	*It is not certain*

_____

num ([1,2,3,6,9,3,4])
realnum ([3.2,4.6,1.1,2.64,100.2])
time (["YESTERDAY","TODAY","TOMORROW"])

_____

```
domains
   bird_list = bird_name *
   bird_name = symbol
   number_list = number *
   number = integer
predicates
     birds(bird_list)
     score(number_list)
clauses
   birds(["sparrow", "robin", "mockingbird" , "thunderbird"]}.
   score ([56,87,63, 89, 91, 62,85] )
```

Operation on lists.

Search of elements in lists.

```
find_it (3, [1,2,3,4,5]).

_____________________
find_it (Head, [Head|_]).

_____________________
find_it (Head, [Head |_].
find_it (Head, [_,Tail]) :- find it (Head, Tail).
```

Division of lists

```
Split (Middle,L,L1,L2).
Split(40, [30,50,20,25,65,95],L1,L2).
split(Middle, [Head|Tail], [Head| LI],L2) :-
Head <= Middle,
split(Middle,Tail,Ll,L2).
split(Middle, [Head|Tail],L1, [Head|L2]) :-
Head > Middle,
split(Middle,Tail, L1,L2),
split (_,[],[],[]).
```

Operation on lists.

Attachment of lists.

L1 =[1,2,3] , **L2 =[4,5] :**
append([], L,L).
append([N |L1],L2,[N |L3]) :- append(LI,L2,L3).
L1= [1,2,3] и **L2= [4,5],**
append([] ,L,L) .
append([],[4,5],_).
append([], [4,5], [4,5]).

Steps of the given process can be presented so:
append([], [4,5], [4,5]) ‘
append([3], [4,5], [3,4,5]|
append([2,3], [4,5], [2,3,4,5])
append([1,2,3], [4,5], [1,2,3,4,*5*])

Operation on lists.

Sorting of lists.

[4,7,3,9] [3,4,7,9],

```
insert_sort([],[]).
insert_sort ([X]| Tail], Sorted__list) :- insert_sort (Tail,Sorted__Tail),
insert(X,Sorted_Tail,Sorted_list).
insert(X|Y Sorted_list] , [Y | Sorted_list 1)) :- asc_order(X,Y), ! ,
insert(X,Sorted_list,Sorted_list1) .
insert(X,Sorted_list,[X|Sorted_list]).
asc_order(X,Y) :- X>Y.
```

_____

```
insert(4, [3,7,9], [3,4,7,9]) .
insert_sort([4,7,3,9],[3,4,7,9]).
```

Operation on lists.

Configuration of data in lists.

findall(Variable_name, Predicate_expression, List_name).

_____

```
domains
        list = real *
        predicates
        football (symbol,real)
        sum_list (list, real, integer).
        average_score
clauses
/* facts (football database) */
        football("Ohio State",116.0).
        football("Michigan",121.0). football("Michigan State",114.0).
        football("Purdue",99.0).
        football("UCLA",122.0).
        average_score :-
        findall(Points,football(_,Points),Point_list), sum_list (Point_list, Sum, Number),
        Average = Sum / Number,
        write("Среднее значение= ", Average).
        sum_list ([],0,0).
        sum_list ([H|T], Sum, Number):-sum_list(T,Sum1,Number1),
        Sum = H + Sum1,
        Number = Number1+ 1.
goal
        average score.
```