LISP SYNTAX:

Atoms, Symbolic atoms, lists

As long as there is no QUOTE mark preceeding the list, LISP will assume that it is dealing with a list containing a function call.

(FIRST <LIST>)

(REST <LIST>)

(DEFUN <NAME> (<PARAMS...>) <FUNCTION BODY...>)

Insert an expression into the front of a list (CONS <EXPRESSION> <LIST>)

Eg: (CONS '(ALL) '(MY MONEY))

list can be constructed from an atomic expression and with only a single element (the atom) (LIST <EXPRESSION> <EXPRESSIONS...>)

(APPEND <LIST> <LISTS...>)

(REVERSE <LIST>)

(LENGTH <LIST>)

(ATOM 'A) ---> T

The predicate `CONSP` tests if an argument is a non-empty list.

`LISTP` also accepts the empty list (NIL) as a list

`(NUMBERP <EXPRESSION >)` and

`(SYMBOLP <EXPRESSION >)`.

The function `EQUAL` with the general calling form `(EQUAL <EXPRESSION1> <EXPRESSION2>)` tests whether two arguments are the same:

`(ENDP <LIST>)`. This function tests whether its argument, which has to be a list, is an empty list, or NIL.

`ZEROP` tests whether the argument, which has to be a number, is the number Zero

`MEMBER`, checks whether an atomically LISP expression is contained in a list. If the LISP expression was found, the tail of the list starting with the LISP expression is returned.

NOT OR AND are also present in LISP.

```
(IF <TEST>
     <THEN-CLAUSE>
     <ELSE-CLAUSE>)
```

```
(COND
    (<TEST1>... <CONSEQUENCE1>)
    (<TEST2>... <CONSEQUENCE2>)
```

The condition part of a T-case is always evaluated if no previous test in `COND` evaluated to `T`!

```
(DEFUN <RECURSIVE-FUNCTION> (<PARAM1> ... <PARAMN>)
(COND (<TERMINATING-CONDITION1> <RESULT1>)
             (<TERMINATING-CONDITIONN> <RESULTN>)
             (<RECURSIVE-CONDITION1> <RECURSIVE-CALL1>)
             (<RECURSIVE-CONDITIONM> <RECURSIVE-CALLM>)))
```

(SETF <VARNAME> <VALUE>)

(MAPCAR <FUNCTION> <LISTE> <LISTS...>)

MAPCAR is a special form. It applies the function successively to each element of argument lists and collects the results in a list

CAR is same as FIRST and CDR is same as REST.

SECOND THIRD LAST returns 2,3and last elements in list.

(SQRT <NUMBER>) gives sqrt of number.