1 Basic Data Structures

1.1 Linked Lists

1.2 Last In - First Out - The Stack

1.3 Dictionaries - The Hash Table

2 Symbols

2.1 `NIL`

`NIL` is general identified with the NULL-pointer. Whenever a NULL-Pointer is encountered, it is interpreted to represent `NIL`.

3 `lambda`

`lambda` is represented by a structure that contains

1. A list of symbols. These symbols will be inserted into the current symbol table with the parameters of the lambda call as values.
2. A pointer to a Block of code to be executed when being called.

Whenever a lambda expression is called with parameters, the following steps will be performed:

1. the parameters' values are inserted into the current lookup table with the symbols of the lambda function. If the symbols already exist within the table, the symbols original values have to be saved.
2. The lambda values code block is executed.
3. The symbols original values are restored.

4 Symbol Tables

1. There is one global Symbol table. This table contains `NIL` etc.
2. Other symbol tables can be created on demand, for example by package. These symbol tables are inserted into the current symbol table.
3. If a package is entered, the current package is pushed onto a stack and the symbol table of the package becomes the new one.

Lookups are performed via

1. the symbol is looked up in the current lookup table.
2. If not found, the symbol table stack is searched for.
3. A name like `part1::part2::part3...` is split up into its parts. Then, `part1` is looked up just as a normal symbol. If it does not resolve to a symbol table, an error is produced. If it resolves to a symbol table, the other parts `part2`, `part3` etc are looked up within this lookup table.