5. Higher Order Procedures

lecture 05 - Lists and recursion.pdf
A Higher Order Procedure either takes a procedure as an argument or returns one as a value.

List Tranformation

Map

Transform lists by applying a function to every element. The procedure provided must be of the indicated format below.
`(map proc lst) -> <List>`;`(proc x) -> y`

Filter

Transform lists by selectively choosing each element according to the procedure. If the given element of the list results in a false value return in the procedure it is filtered out of the list.
`(filter proc lst) -> <List>'` `(proc x) -> #t or #f`

Reduce

Transforms lists by using a procedure to iteratively process each element of the list.
`(reduce proc init lst) -> <Value>`
`(proc accumulated element) -> new_accumulated`

- proc: A function that takes two arguments (the current accumulated value and the current element of the list) and returns a new accumulated value.
- init: The initial value for the accumulation, setting the starting point.
- lst: The list to be processed, where each element is passed along with the accumulated value to the procedure.

This process continues until the list is exhausted, resulting in a single, final accumulated value.

Example:

```
(reduce + 0 '(1 2 3 4 5)) ==> 15
```