Structure and Interpretation of Computer Programs

R6A
Data Abstraction

Source §2

1. `pair(x, y)`: makes a pair from `x` and `y`
2. `head(p)`: returns the head (first component) of the pair `p`
3. `tail(p)`: returns the tail (second component) of the pair `p`
4. `list(x1, x2, x3, ..., xn)`: returns a list with `n` elements. The first element is `x1`, the second `x2`, etc.
5. `length(xs)`: returns the length of the list `xs`
6. `list_ref(xs, n)`: returns element of list `xs` at position `n`, where the first element is at position 0.

Problems:

1. Draw the box-and-pointer diagrams for the results of evaluating the following programs. Also write the data structures in box notation and in list notation.

   (a) `pair(1, 2);`
   (b) `pair(1, pair(3, pair(5, null)));`
   (c) `pair(pair(pair(3, 2), pair(1, 0)), null);`
   (d) `pair(0, list(1, 2));`

2. Write Source §2 programs that evaluate to values that are printed out in the Source REPL as follows:

   (a) `[1, [2, [3, null]]]`

   (b) `[1, [2, 3]]`

   (c) `[[1, [2, null]], [[3, [4, null]], [[5, [6, null]], null]]]`

3. Write programs that only use applications of `head` and `tail` and the name `lst` so that the result is 4:

   (a) `const lst = list(7, 6, 5, 4, 3, 2, 1);`
   (b) `const lst = list(list(7), list(6, 5, 4), list(3, 2), 1);`
   (c) `const lst = list(7, list(6, list(5, list(4, list(3, list(2, list(1)))))));`
   (d) `const lst = list(7, list(list(list(6, 5, list(list(4)), 3), 2) ), 1);`

   Note: The key skill for this question is to translate an expression into a box-and-pointer diagram and to systematically traverse the data structure.