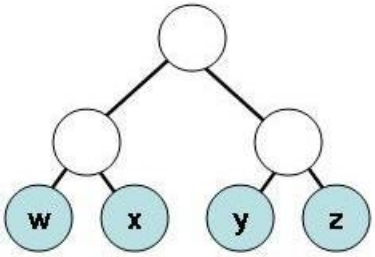
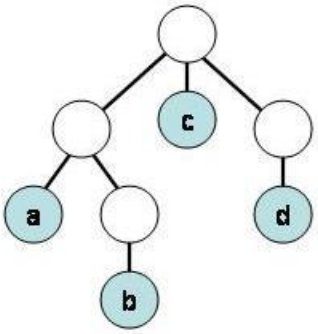
FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE - CS161
Fall 2013

Assignment 2 - Due 11:59pm Thursday, October 17

The first two problems are to implement two brute-force search algorithms: depth-first and depth-first iterative-deepening. The search trees will be represented as lists in which a leaf node is represented by an atom, and a non-leaf node is represented by a list of its child nodes. For example, the list ((W X) (Y Z)) represents the complete two-level binary tree shown below on the left, and the list ((A (B)) C (D)) represents the tree shown below on the right.



((w x) (y z))



((a (b)) c (d))

The third problem is to implement a depth-first solver for the missionary-cannibal problem discussed in class using a code skeleton.

You are restricted to using the following functions, predicates, and operators introduced in class: null, atom, append, quote ['], car, cdr [cadadr, etc.], first, second [third, etc.], rest, cons, list, listp, null, cond, equal, defun, let, let*, and, or, not, =, +, -, *, /, >, <. Note: you are not permitted to use setq or last. Submit **via Courseweb** the code for all problems in a file named**: "hw2.lsp"**. Submit **via Courseweb** the test cases in a file named "**test.txt"**.

1. Write a single pure LISP function, called DFS, that performs a depth-first search of a tree. The function should take a single argument that is the list representation of the tree, and return a single, top-level list of the terminal nodes in the order they would be visited by a left-to-right depth-first search. For example, (dfs '((A (B)) C (D))) would return (A B C D). Do not use any auxiliary functions.

2. Write a *set* of pure LISP functions that implement depth-first iterative-deepening. Your top-level function, called DFID, should take two arguments, the list representation of the tree, and an integer representing the maximum depth of the tree, and return a single top-level list of the terminal nodes in the order that they would be visited by a left-to-right depth-first iterative-deepening search. Note that those nodes that are visited in

multiple iterations will appear multiple times in the output list. For example, (dfid '((A (B)) C (D)) 3) would return (C A C D A B C D).

Each of these functions must work for trees of arbitrary depth and branching factor, and hence you may not assume any a priori upper bound on these parameters. Be sure to exhibit sufficient test cases to convince yourself and us that your programs work in general. Include at least the examples above, as well as the list (A (B C) (D) (E (F G))).

3. Implement a depth-first solver for the missionary-cannibal problem that was described in class. On Courseweb, you can find a file called "hw2_skeleton.lsp"; to implement this solver, implement the functions in it as described in their associated comments. DO NOT CHANGE THE FUNCTION NAMES OR PARAMETERS. Also do not write any additional functions.